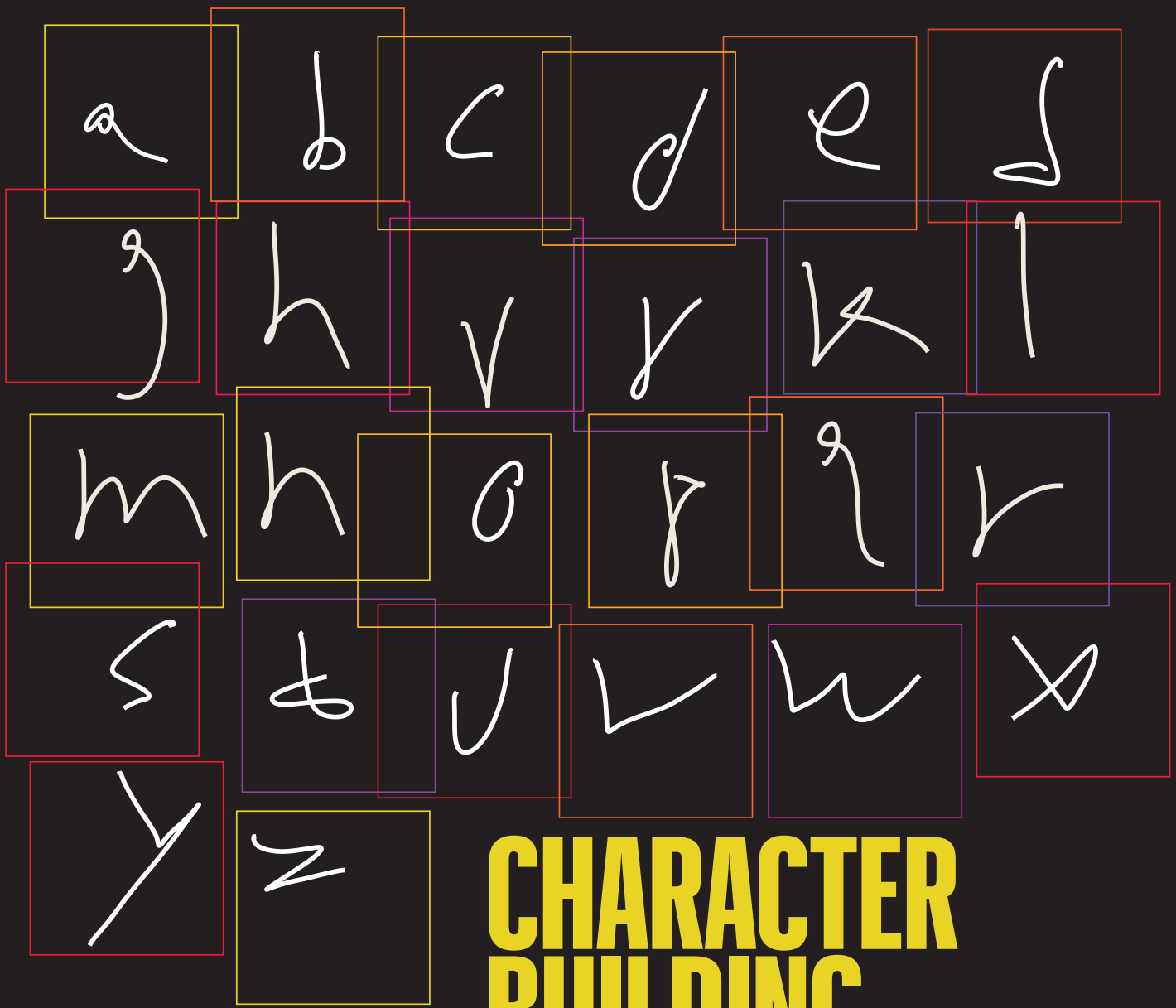
Table of Contents

A “single file with everything”, except with only Stanford, HHMI, Nature Podcast, Science & NPR articles.

The international journal of science / 13 May 2021

nature



CHARACTER BUILDING

Brain–computer interface translates thoughts of handwriting into typed text

Cover line
Cover blurb goes in here on three long lines of loveliness

Cover line
Cover blurb goes in here on three long lines of loveliness

Cover line
Cover blurb goes in here on three long lines of loveliness

Vol. 593, No. 7858
£10.00 nature.com

News & views

Neuroscience

Neural interface translates thoughts into type

Pavithra Rajeswaran & Amy L. Orsborn

A neural interface has been developed that could enable people with paralysis to type faster than they could using other technologies, by directly translating attempts at handwriting into text. **See p.249**

We can think much faster than we can communicate – a fact that many of us feel aware of as we struggle with our smartphone keyboards. For people with severe paralysis, this information bottleneck is much more extreme. Willett *et al.*[1] report on page 249 the development of a brain–computer interface (BCI) for typing that could eventually let people with paralysis communicate at the speed of their thoughts.

Commercially available assistive typing devices predominantly rely on the person using the device being able to make eye movements or deliver voice commands. Eye-tracking keyboards can let people with paralysis type at around 47.5 characters per minute[2], slower than the 115-per-minute speeds achieved by people without a comparable injury. However, these technologies do not work for people whose paralysis impairs eye movements or vocalization. And the technology has limitations. For instance, it is hard to reread an e-mail, so that you can compose your reply, while you are typing with your eyes.

By contrast, BCIs restore function by deciphering patterns of brain activity. Such interfaces have successfully restored simple movements – such as reaching for and manipulating large objects – to people with paralysis[3–7]. By directly tapping into neural processing, BCIs hold the tantalizing promise of seamlessly restoring function to a wide range of people.

But, so far, BCIs for typing have been unable to compete with simpler assistive technologies such as eye-trackers. One reason is that typing is a complex task. In English, we select from 26 letters of the Latin alphabet. Building a classification algorithm to predict which letter a user wants to choose, on the basis of their neural activity, is challenging, so BCIs have solved typing tasks indirectly. For instance, non-invasive BCI spellers present several sequential visual cues to the user and analyse the neural responses to all cues to determine the desired letter[8]. The most successful invasive BCI (iBCI; one that involves implanting an electrode into the brain) for typing allowed users to control a cursor to select keys, and achieved speeds of 40 characters per minute[6]. But these iBCIs, like non-invasive eye-trackers, occupy the user's visual attention and do not provide notably faster typing speeds.

Willett and colleagues developed a different approach, which directly solves the typing task in an iBCI and thereby leapfrogs far beyond past devices, in terms of both performance and functionality. The approach involves decoding letters as users imagine writing at their own pace (Fig. 1).

Such an approach required a classification algorithm that predicts which of 26 letters or 5 punctuation marks a user with paralysis is trying to write – a challenging feat when the attempts cannot be observed and occur whenever the user chooses. To overcome this challenge, Willett *et al.* first repurposed another type of algorithm – a machine-learning algorithm originally developed for speech recognition. This allowed them to estimate, on the basis of neural activity alone, when a user started attempting to write a character. The pattern of neural activity generated each time their study participant imagined a given character was remarkably consistent. From this information, the group produced a labelled data set that contained the neural-activity patterns corresponding to each character. They used this data set to train the classification algorithm.

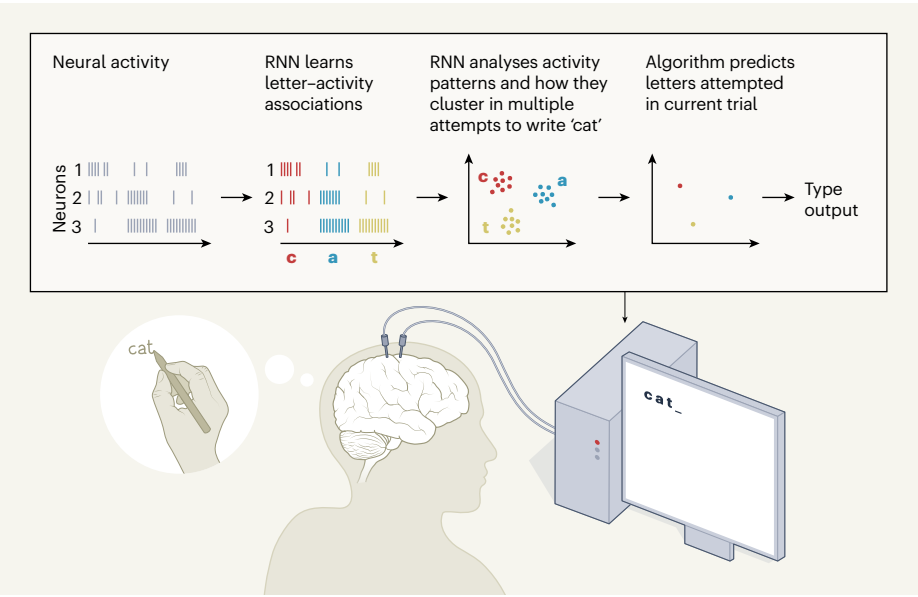


Figure 1 | A brain–computer interface (BCI) for typing. Willett *et al.*[1] have developed a BCI that enables a person with paralysis to type, by translating the neural activity produced from imagined attempts at handwriting into text on the computer screen. As a simplified description, electrodes implanted into the brain measure the activity of many neurons as the user imagines writing each letter (lines indicate time points at which each neuron fires). A deep-learning model called a recurrent neural network (RNN) learns the neural activity patterns produced from each character, and analyses how these activity patterns relate across multiple trials, generating cluster plots. This information is used to by an algorithm to predict the letters being imagined by the participant in the current trial, and the prediction is translated into a typographic output. (Figure adapted from Fig. 2a of ref. 1.)

To achieve accurate classification in such a high-dimensional space, Willett and colleagues' classification algorithm used current machine-learning methods, along with a type of artificial neural network called a recurrent neural network (RNN), which is especially good at predicting sequential data. Harnessing the power of RNNs requires ample training data, but such data are limited in neural interfaces, because few users want to imagine writing for hours on end. The authors solved this problem using an approach known as data augmentation, in which neural activity patterns previously generated by the participant are used to produce artificial sentences on which to train the RNN. They also expanded their training data by introducing artificial variability into the patterns of neural activity, to mimic changes that occur naturally in the human brain. Such variability can make RNN BCIs more robust[9].

Thanks to these methods, Willett and colleagues' algorithm provided impressively accurate classification, picking the correct character 94.1% of the time. By including predictive-language models (similar to those that drive auto-correct functions on a smartphone), they further improved accuracy to 99.1%. The participant was able to type accurately at a speed of 90 characters per minute — a twofold improvement on his performance with past iBCIs.

This study's achievements, however, stem from more than machine learning. A decoder's performance is ultimately only as good as the data that are fed into it. The researchers found that neural data associated with attempted handwriting are particularly well-suited for typing tasks and classification. In fact, handwriting could be classified quite well even with simpler, linear algorithms, suggesting that the neural data themselves played a large part in the success of the authors' approach.

By simulating how the classification algorithm performed when tested with different types of neural activity, Willett *et al.* made a key insight — neural activity during handwriting has more temporal variability between characters than does neural activity when users attempt to draw straight lines, and this variablility actually makes classification easier. This knowledge should inform future BCIs. Perhaps counter-intuitively, it might be advantageous to decode complex behaviours rather than simple ones, particularly for classification tasks.

Willett and co-workers' study begins to deliver on the promise of BCI technologies. iBCIs will need to provide tremendous performance and usability benefits to justify the expense and risks associated with implanting electrodes into the brain. Importantly, typing speed is not the only factor that will determine whether the technology is adopted — the longevity and robustness of the approach also require analysis. The authors present promising evidence that their algorithms will perform well with limited training data, but further research will probably be required to enable the device to maintain performance over its lifetime as neural activity patterns change. It will also be crucial to conduct studies to test whether the approach can be generalized for other users, and for settings outside the laboratory.

Another question is how the approach will scale and translate to other languages. Willett and colleagues' simulations highlight that several characters of the Latin alphabet are written similarly (r, v and u, for instance), and so are harder to classify than are others. One of us (P.R.) speaks Tamil, which has 247, often very closely related, characters, and so might be much harder to classify. And the question of translation is particularly pertinent for languages that are not yet well represented in machine-learning predictive-language models.

Although much work remains to be done, Willett and co-workers' study is a milestone that broadens the horizon of iBCI applications. Because it uses machine-learning methods that are rapidly improving, plugging in the latest models offers a promising path for future improvements. The team is also making its data set publicly available, which will accelerate advances. The authors' approach has brought neural interfaces that allow rapid communication much closer to a practical reality.

Pavithra Rajeswaran and **Amy L. Orsborn** are in the Department of Bioengineering, University of Washington, Seattle, Washington 98195, USA. **A.L.O.** is also in the Department of Electrical and Computer Engineering, and at the Washington National Primate Research Center, University of Washington.
e-mail: aorsborn@uw.edu

1. Willett, F. R., Avansino, D. T., Hochberg, L. R., Henderson, J. M. & Shenoy, K. V. *Nature* **593**, 249–254 (2021).
2. Mott, M. E., Williams, S., Wobbrock, J. O. & Morris, M. R. in *Proc. 2017 CHI Conf. Human Factors in Computing Systems* 2558–2570 (ACM, 2017).
3. Hochberg, L. R. *et al. Nature* **442**, 164–171 (2006).
4. Hochberg, L. R. *et al. Nature* **485**, 372–375 (2012).
5. Collinger, J. L. *et al. Lancet* **381**, 557–564 (2013).
6. Pandarinath, C. *et al. eLife* **6**, e18554 (2017).
7. Ajiboye, A. B. *et al. Lancet* **389**, 1821–1830 (2017).
8. Rezeika, A. *et al. Brain Sci.* **8**, 57 (2018).
9. Sussillo, D., Stavisky, S. D., Kao, J. C., Ryu, S. I. & Shenoy, K. V. *Nature Commun.* **7**, 13749 (2016).

Atmospheric chemistry

How ant acid forms in the atmosphere

Joost de Gouw & Delphine Farmer

Known sources of formic acid could not explain the observed atmospheric concentrations of this compound. The discovery of a previously unknown pathway that generates formic acid in the atmosphere resolves this discrepancy. **See p.233**

Formic acid is one of the simplest and most abundant organic molecules in Earth's atmosphere, but its sources have been poorly understood for many years. Laboratory and field studies[1–3] have shown that most formic acid is not emitted directly from sources, but is produced by chemical reactions in the atmosphere. However, the chemistry responsible has been a mystery. On page 233, Franco *et al.*[4] report that formic acid could be formed by a mechanism that starts with formaldehyde reacting with water in cloud droplets.

The word 'formic' derives from *formica*, the Latin word for ant, and the compound is indeed released from ant hills[5]. Other, and larger, emission sources include vegetation, biomass burning[6] and fossil-fuel combustion[7]. However, the combined emissions from known sources are too small to explain the concentrations of formic acid in the atmosphere, and several studies have concluded that formation in the atmosphere is a much bigger contributor (see ref. 1, for example).

Levels of formic acid can be measured by mass spectrometry and optical spectroscopy, and from satellite instruments, so there is excellent information about the distribution of this compound in the atmosphere. Observations have shown that atmospheric concentrations of formic acid increase rapidly in urban[8] and forest[9] air during the day. However, researchers have been unable to identify the chemical reactions responsible for this increase. Detailed studies that considered all of the possible known chemical pathways could explain only a fraction of formic acid produced, both in polluted and remote regions[3,9], and so the search for alternative

Article

High-performance brain-to-text communication via handwriting

https://doi.org/10.1038/s41586-021-03506-2

Received: 2 July 2020

Accepted: 26 March 2021

Published online: 12 May 2021

Check for updates

Francis R. Willett[1,2,3]✉, Donald T. Avansino[1], Leigh R. Hochberg[4,5,6,7], Jaimie M. Henderson[2,8,9,12] & Krishna V. Shenoy[1,3,8,9,10,11,12]

Brain–computer interfaces (BCIs) can restore communication to people who have lost the ability to move or speak. So far, a major focus of BCI research has been on restoring gross motor skills, such as reaching and grasping[1–5] or point-and-click typing with a computer cursor[6,7]. However, rapid sequences of highly dexterous behaviours, such as handwriting or touch typing, might enable faster rates of communication. Here we developed an intracortical BCI that decodes attempted handwriting movements from neural activity in the motor cortex and translates it to text in real time, using a recurrent neural network decoding approach. With this BCI, our study participant, whose hand was paralysed from spinal cord injury, achieved typing speeds of 90 characters per minute with 94.1% raw accuracy online, and greater than 99% accuracy offline with a general-purpose autocorrect. To our knowledge, these typing speeds exceed those reported for any other BCI, and are comparable to typical smartphone typing speeds of individuals in the age group of our participant (115 characters per minute)[8]. Finally, theoretical considerations explain why temporally complex movements, such as handwriting, may be fundamentally easier to decode than point-to-point movements. Our results open a new approach for BCIs and demonstrate the feasibility of accurately decoding rapid, dexterous movements years after paralysis.

Previous BCI studies have shown that the motor intention for gross motor skills, such as reaching, grasping or moving a computer cursor, remains neurally encoded in the motor cortex after paralysis[1–7]. However, it is still unknown whether the neural representation for a rapid and highly dexterous motor skill, such as handwriting, also remains intact. We tested this by recording neural activity from two microelectrode arrays in the hand 'knob' area of the precentral gyrus (a premotor area)[9,10] while our BrainGate study participant, T5, attempted to handwrite individual letters and symbols (Fig. 1a). T5 has a high-level spinal cord injury and was paralysed from the neck down; his hand movements were entirely non-functional and limited to twitching and micromotion. We instructed T5 to 'attempt' to write as if his hand were not paralysed, while imagining that he was holding a pen on a piece of ruled paper.

Neural representation of handwriting

To visualize the neural activity (multiunit threshold crossing rates) recorded during attempted handwriting, we used principal components analysis to display the top three neural dimensions that contain the most variance (Fig. 1b). The neural activity appeared to be strong and repeatable, although the timing of its peaks and valleys varied across trials, potentially owing to fluctuations in writing speed. We used a time-alignment technique to remove temporal variability[11], which revealed notably consistent underlying patterns of neural activity that are unique to each character (Fig. 1c). To ascertain whether the neural activity encoded the pen movements that are needed to draw each character's shape, we attempted to reconstruct each character by linearly decoding the pen-tip velocity from the trial-averaged neural activity (Fig. 1d). Readily recognizable letter shapes confirmed that pen-tip velocity is robustly encoded. The neural dimensions that represented pen-tip velocity accounted for 30% of the total neural variance.

Next, we used a nonlinear dimensionality reduction method (*t*-distributed stochastic neighbour embedding; t-SNE) to produce a two-dimensional (2D) visualization of each single trial's neural activity recorded after the 'go' cue was given (Fig. 1e). The t-SNE visualization revealed tight clusters of neural activity for each character and a predominantly motoric encoding in which characters that are written similarly are closer together. Using a *k*-nearest-neighbour classifier applied offline to the neural activity, we could classify the characters with 94.1% accuracy (95% confidence interval (CI) = [92.6, 95.8]). Together, these results suggest that, even years after paralysis, the neural representation of handwriting in the motor cortex is probably strong enough to be useful for a BCI.

[1]Howard Hughes Medical Institute at Stanford University, Stanford, CA, USA. [2]Department of Neurosurgery, Stanford University School of Medicine, Stanford, CA, USA. [3]Department of Electrical Engineering, Stanford University, Stanford, CA, USA. [4]VA RR&D Center for Neurorestoration and Neurotechnology, Rehabilitation R&D Service, Providence VA Medical Center, Providence, RI, USA. [5]School of Engineering, Brown University, Providence, RI, USA. [6]Carney Institute for Brain Science, Brown University, Providence, RI, USA. [7]Center for Neurotechnology and Neurorecovery, Department of Neurology, Massachusetts General Hospital, Harvard Medical School, Boston, MA, USA. [8]Wu Tsai Neurosciences Institute, Stanford University, Stanford, CA, USA. [9]Bio-X Institute, Stanford University, Stanford, CA, USA. [10]Department of Bioengineering, Stanford University, Stanford, CA, USA. [11]Department of Neurobiology, Stanford University, Stanford, CA, USA. [12]These authors jointly supervised this work: Jaimie M. Henderson, Krishna V. Shenoy. ✉e-mail: fwillett@stanford.edu

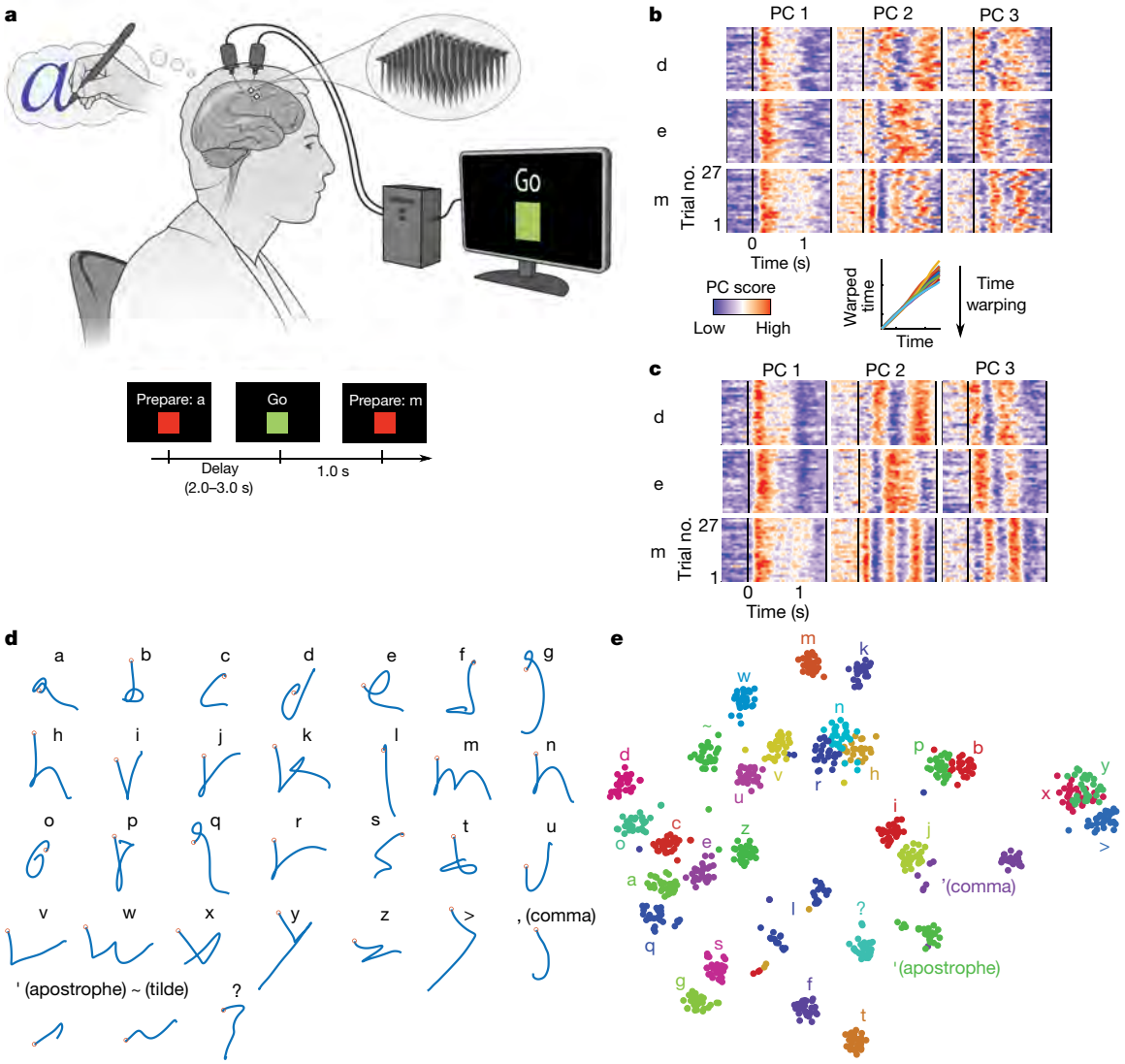


Fig. 1 | Neural representation of attempted handwriting. a, To assess the neural representation of attempted handwriting, participant T5 attempted to handwrite each character one at a time, following the instructions given on a computer screen (bottom panels depict what is shown on the screen, following the timeline). Credit: drawing of the human silhouette created by E. Woodrum. **b**, Neural activity in the top 3 principal components (PCs) is shown for three example letters (d, e and m) and 27 repetitions of each letter (trials). The colour scale was normalized within each panel separately for visualization. **c**, Time-warping the neural activity to remove trial-to-trial changes in writing speed reveals consistent patterns of activity unique to each letter. In the inset above **c**, example time-warping functions are shown for the letter 'm' and lie relatively close to the identity line (the warping function of each trial is plotted with a different coloured line). **d**, Decoded pen trajectories are shown for all 31 tested characters. Intended 2D pen-tip velocity was linearly decoded from the neural activity using cross-validation (each character was held out), and decoder output was denoised by averaging across trials. Orange circles denote the start of the trajectory. **e**, A 2D visualization of the neural activity made using t-SNE. Each circle is a single trial (27 trials are shown for each of 31 characters).

Decoding handwritten sentences

Next, we tested whether we could decode complete handwritten sentences in real time, thus enabling an individual with tetraplegia to communicate by attempting to handwrite their intended message. To do so, we trained a recurrent neural network (RNN) to convert the neural activity into probabilities describing the likelihood of each character being written at each moment in time (Fig. 2a, Extended Data Fig. 1). These probabilities could either be thresholded in a simple way to emit discrete characters, which we did for real-time decoding ('raw online output', Fig. 2a), or processed more extensively by a large-vocabulary language model to simulate an autocorrect feature, which we applied offline ('offline output from a language model', Fig. 2a). We used the limited set of 31 characters shown in Fig. 1d, consisting of the 26 lower-case letters of the alphabet, together with commas, apostrophes, question marks, full stops (written by T5 as a tilde symbol; '~') and spaces (written by T5 as a greater-than symbol; '>'). The '~' and '>' symbols were chosen to make full stops and spaces easier to detect. T5 attempted to write each character in print (not cursive), with each character printed on top of the previous one.

To collect training data for the RNN, we recorded neural activity while T5 attempted to handwrite complete sentences at his own pace, following instructions on a computer monitor. Before the first day of real-time evaluation, we collected a total of 242 sentences across 3 pilot days that were combined to train the RNN. On each subsequent day of real-time testing, additional training data were collected to recalibrate the RNN before evaluation, yielding a combined total of 572 training sentences by the last day (comprising 7.6 hours and 31,472 characters). To train the RNN, we adapted neural network methods in automatic speech recognition[12–14] to overcome two key challenges: (1) the time that each letter was written in the training data was unknown (as T5's hand was paralysed), making it challenging to apply supervised learning

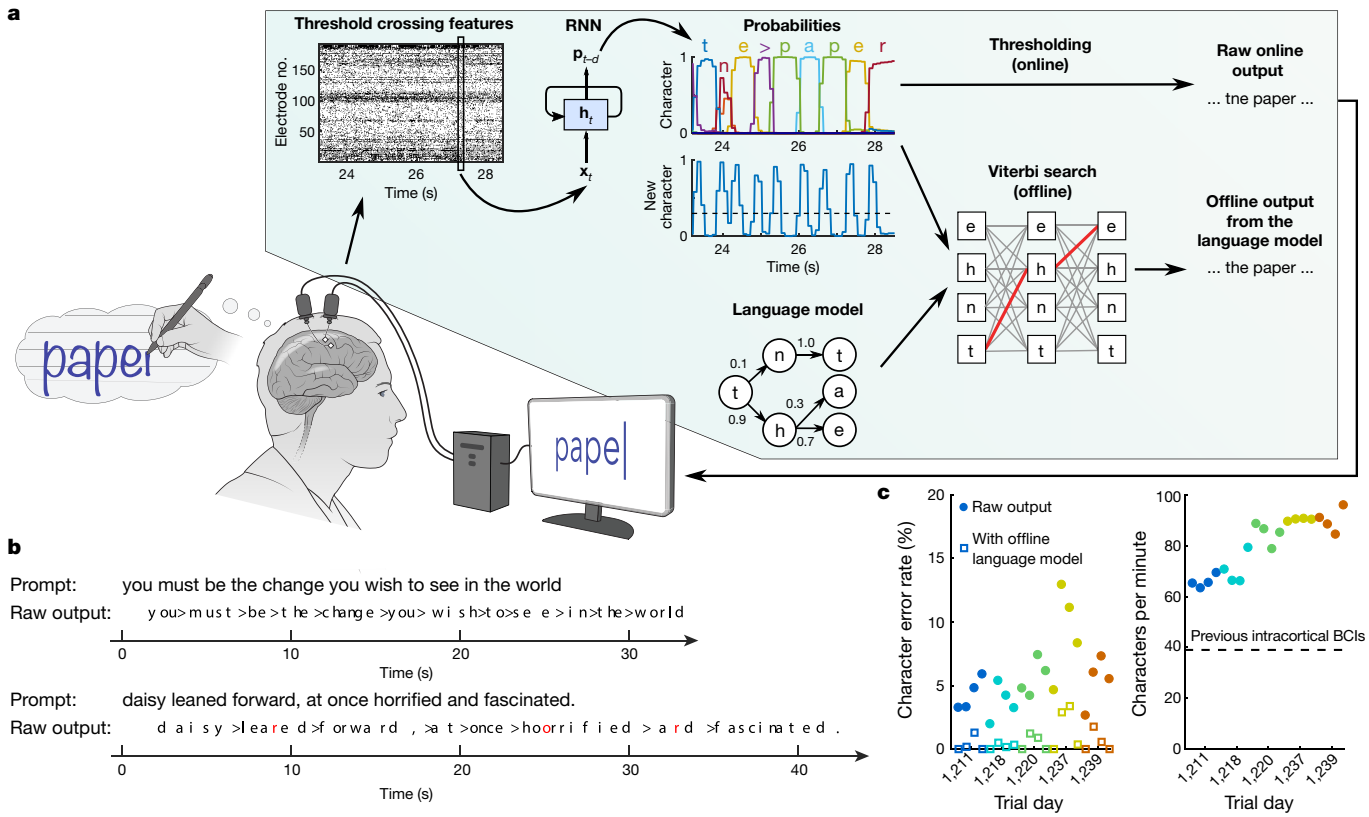


Fig. 2 | Neural decoding of attempted handwriting in real time. a, Diagram of the decoding algorithm. First, the neural activity (multiunit threshold crossings) was temporally binned and smoothed on each electrode (20-ms bins). Then, an RNN converted this neural population time series ($\mathbf{x}_t$) into a probability time series ($\mathbf{p}_{t-d}$) describing the likelihood of each character and the probability of any new character beginning. The RNN had a one second output delay (d), giving it time to observe each character fully before deciding its identity. Finally, the character probabilities were thresholded to produce the 'raw online output' for real-time use (when the 'new character' probability crossed a threshold at time t, the most likely character at time $t + 0.3$ s was emitted and shown on the screen). In an offline retrospective analysis, the character probabilities were combined with a large-vocabulary language model to decode the most likely text that the participant wrote (using a custom 50,000-word bigram model). Credit: drawing of the human silhouette created by E. Woodrum. **b**, Two real-time example trials are shown, demonstrating the ability of the RNN to decode readily understandable text on sentences on which it was never trained. Errors are highlighted in red and spaces are denoted with '>'. **c**, Error rates (edit distances) and typing speeds are shown for 5 days, with 4 blocks of 7–10 sentences each (each block is indicated with a single circle and coloured according to the trial day). The speed is more than double that of the next-fastest intracortical BCI[7], which is indicated with the dashed line.

techniques; and (2) the dataset was limited in size compared to typical RNN datasets, making it difficult to prevent overfitting to the training data (see Supplementary Methods, Extended Data Figs. 2, 3).

We evaluated the performance of the RNN over a series of 5 days, each day containing 4 evaluation blocks of 7–10 sentences that the RNN was never trained on (thus ensuring that the RNN could not overfit to those sentences). T5 copied each sentence from an on-screen prompt, attempting to handwrite it letter by letter, while the decoded characters appeared on the screen in real time as they were detected by the RNN (Supplementary Videos 1, 2, Extended Data Table 1). Characters appeared after they were completed by T5 with a short delay (estimated to be 0.4–0.7 s). The decoded sentences were quite legible ('raw output', Fig. 2b). Notably, typing speeds were high, plateauing at 90 characters per minute with a mean character error rate of 5.4% (averaged across all four blocks on the final day) (Fig. 2c). As there was no 'backspace' function implemented, T5 was instructed to continue writing if any decoding errors occurred.

When a language model was used to autocorrect errors offline, error rates decreased considerably (Fig. 2c, Table 1). The character error rate decreased to 0.89% and the word error rate decreased to 3.4% averaged across all days, which is comparable to state-of-the-art speech recognition systems with word error rates of 4–5%[14,15], putting it well within the range of usability. Finally, to probe the limits of possible decoding performance, we trained a new RNN offline using all available sentences to process the entire sentence in a non-causal way (comparable to other BCI studies[16,17]). Accuracy was extremely high in this regime (0.17% character error rate), indicating a high potential ceiling of performance, although this decoder would not be able to provide letter-by-letter feedback to the user.

Next, to evaluate performance in a less restrained setting, we collected two days of data in which T5 used the BCI to freely type answers to open-ended questions (Supplementary Video 3, Extended Data Table 2). The results confirm that high performance can also be achieved when the user writes self-generated sentences as opposed to copying on-screen prompts (73.8 characters per minute, with a character error rate of 8.54% in real time and 2.25% with a language model). To our knowledge, the previous record for free typing in intracortical BCIs is 24.4 correct characters per minute[7].

Daily decoder retraining

Following standard practice[1,2,4,5,18], we retrained our handwriting decoder each day before evaluating it, with the help of 'calibration' data collected at the beginning of each day. Retraining helps to account

Table 1 | Mean character and word error rates (with 95% CIs) for the handwriting BCI across all 5 days

	Character error rate [95% CI]	Word error rate [95% CI]
Raw online output	5.9% [5.3, 6.5]	25.1% [22.5, 27.4]
Online output + offline language model	0.89% [0.61, 1.2]	3.4% [2.5, 4.4]
Offline bidirectional RNN + language model	0.17% [0, 0.36]	1.5% [0, 3.2]

'Raw online output' is what was decoded online (in real time). 'Online output + offline language model' was obtained by applying a language model retrospectively to what was decoded online (to simulate an autocorrect feature). 'Offline bidirectional RNN + language model' was obtained by retraining a bidirectional (acausal) decoder offline using all available data, in addition to applying a language model. Word error rates can be much higher than character error rates because a word is considered incorrect if any character in that word is wrong.

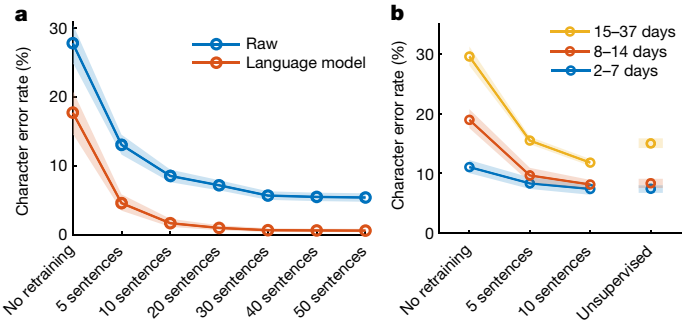


Fig. 3 | Performance remains high when daily decoder retraining is shortened (or unsupervised). a, To account for changes in neural activity that accrue over time, we retrained our handwriting decoder each day before evaluating it. Here, we simulated offline how decoding performance would have changed if fewer than the original 50 calibration sentences were used. Lines show the mean error rate across all data and shaded regions indicate 95% CIs. **b**, Copy-typing data from eight sessions were used to assess whether fewer calibration data are required if sessions occur closer in time. All session pairs (X, Y) were considered. Decoders were first initialized using training data from session X and earlier, and then evaluated on session Y under different retraining methods (no retraining, retraining with limited calibration data, or unsupervised retraining). Lines show the average raw error rate and shaded regions indicate 95% CIs.

for changes in neural recordings that accrue over time, which might be caused by neural plasticity or electrode array micromotion. Ideally, to reduce the burden on the user, minimal or no calibration data would be required. In a retrospective analysis of the copy-typing data reported above in Fig. 2, we assessed whether high performance could still have been achieved using fewer than the original 50 calibration sentences per day (Fig. 3a). We found that 10 sentences (8.7 min) were enough to achieve a raw error rate of 8.5% (1.7% with a language model), although 30 sentences were needed to match the raw online error rate of 5.9%.

However, our copy-typing data were collected over a 28-day time span, possibly allowing larger changes in neural activity to accumulate. Using further offline analyses, we assessed whether sessions that are more closely spaced reduce the need for calibration data (Fig. 3b). We found that when only 2–7 days passed between sessions, performance was reasonable with no decoder retraining (11.1% raw error rate, 1.5% with a language model), as might be expected from previous work showing the short-term stability of neural recordings[19–21]. Finally, we tested whether decoders could be retrained in an unsupervised manner by using a language model to error-correct and retrain the decoder, thus bypassing the need to interrupt the user for calibration (by enabling automatic recalibration during normal use). Encouragingly, unsupervised retraining achieved a raw error rate of 7.3% (0.84% with a language model) when sessions were separated by a time span of 7 days or less.

Ultimately, whether decoders can be successfully retrained with minimal recalibration data depends on how quickly the neural activity changes over time. We assessed the stability of the neural patterns associated with each character and found high short-term stability (mean correlation of 0.85 when 7 days apart or less), and neural changes that seemed to accumulate at a steady and predictable rate (Extended Data Fig. 4). These results are promising for clinical viability, as they suggest that unsupervised decoder retraining, combined with more-limited supervised retraining after longer periods of inactivity, may be sufficient to achieve high performance. Nevertheless, future work must confirm this online, as offline simulations are not always predictive of online performance.

Temporal variety improves decoding

To our knowledge, 90 characters per minute is the highest typing rate that has yet been reported for any type of BCI (see 'Discussion'). For intracortical BCIs, the best-performing method has been point-and-click typing with a 2D computer cursor, which peaks at 40 correct characters per minute[7] (see Supplementary Video 4 for a direct comparison). The speed of point-and-click BCIs is limited primarily by decoding accuracy. During parameter optimization, the cursor gain is increased so as to increase typing rate, until the cursor moves so quickly that it becomes uncontrollable owing to decoding errors[22]. We therefore asked how handwriting movements could be decoded more than twice as fast, with similar levels of accuracy.

We theorize that handwritten letters may be easier to distinguish from each other than point-to-point movements, as letters have more variety in their spatiotemporal patterns of neural activity than do straight-line movements. To test this theory, we analysed the patterns of neural activity associated with 16 straight-line movements and 16 letters that required no lifting of the pen off the page, both performed by T5 with attempted handwriting (Fig. 4a, b).

First, we analysed the pairwise Euclidean distances between each neural activity pattern. We found that the nearest-neighbour distances for each movement were 72% larger for characters compared to straight lines (95% CI = [60%, 86%]), making it less likely for a decoder to confuse two nearby characters (Fig. 4c). To confirm this, we simulated the classification accuracy for each set of movements as a function of neural noise (Fig. 4d), which showed that characters are easier to classify than straight lines.

To gain insight into what might be responsible for the relative increase in nearest-neighbour distances for characters, we examined the spatial and temporal dimensionality of the neural patterns. Spatial and temporal dimensionality were estimated using the 'participation ratio' of the principal component analysis (PCA) eigenvalue spectrum, which quantifies approximately how many spatial or temporal dimensions are required to explain 80% of the variance in the patterns of neural activity[23]. We found that the spatial dimensionality was only modestly larger for characters (1.24 times larger; 95% CI = [1.19, 1.30]), but that the temporal dimensionality was much greater (2.65 times larger; 95% CI = [2.58, 2.72]), suggesting that the increased variety of temporal patterns in letter writing drives the increased separability of each movement (Fig. 4e).

To illustrate how increased temporal dimensionality can make movements more distinguishable, we constructed a toy model with four movements and two neurons, with the neural activity constrained to lie along a single dimension (Fig. 4f, g). Simply by allowing the trajectories to change in time (Fig. 4g), the nearest-neighbour distance between the neural trajectories can be increased, resulting in an increase in classification accuracy when noise levels are large enough (Fig. 4h). Although neural noise in the toy model was assumed to be independent white noise, we found that these results also hold for noise that

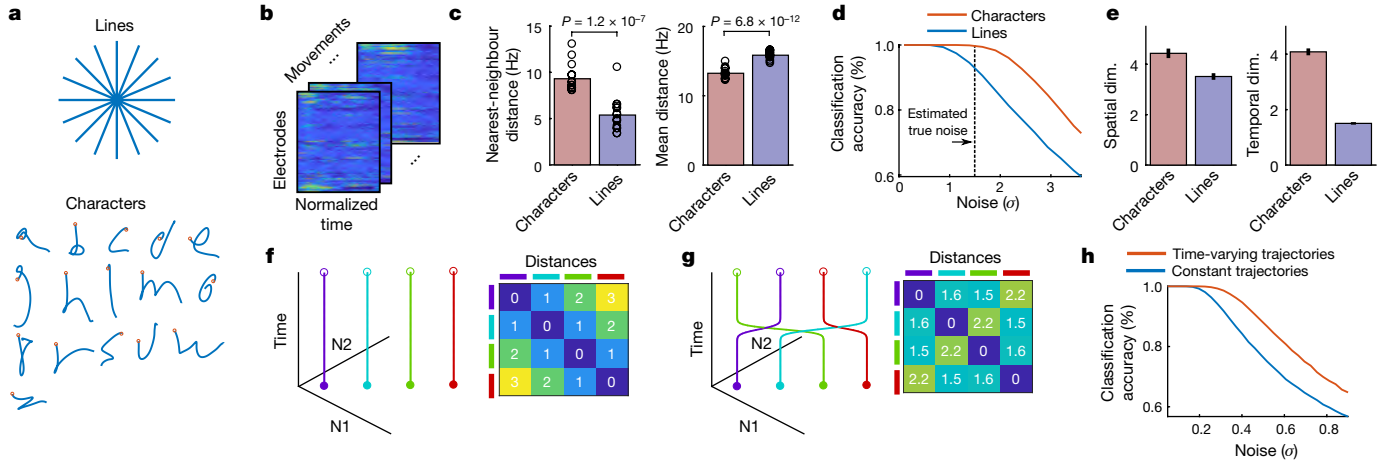


Fig. 4 | Increased temporal variety can make movements easier to decode. **a**, We analysed the spatiotemporal patterns of neural activity corresponding to 16 handwritten characters (1 s in duration) versus 16 handwritten straight-line movements (0.6 s in duration). **b**, Spatiotemporal neural patterns were found by averaging over all trials for a given movement (after time-warping to align the trials in time)[11]. Neural activity was resampled to equalize the duration of each set of movements, resulting in a 192 × 100 matrix for each movement (192 electrodes and 100 time steps). **c**, Pairwise Euclidean distances between neural patterns were computed for each set, revealing larger nearest-neighbour distances (but not mean distances) for characters. Each circle represents a single movement and bar heights show the mean. **d**, Larger nearest-neighbour distances made the characters easier to classify than straight lines. The noise is in units of standard deviations and matches the scale of the distances in **c**. **e**, The spatial dimensionality (dim.) was similar for characters and straight lines, but the temporal dimensionality was more than twice as high for characters, suggesting that more temporal variety underlies the increased nearest-neighbour distances and better classification performance. Error bars show 95% CIs. Dimensionality was quantified using the participation ratio. **f**–**h**, A toy example to give intuition for how increased temporal dimensionality can make neural trajectories more separable. Four neural trajectories are depicted (N1 and N2 are two hypothetical neurons, the activity of which is constrained to a single spatial dimension, the unity diagonal). Allowing the trajectories to vary in time by adding one bend, which increases the temporal dimensionality from 1 (**f**) to 2 (**g**), enables larger nearest-neighbour distances and better classification (**h**).

is correlated across time and neurons (Extended Data Fig. 5, Supplementary Note 1).

These results suggest that time-varying patterns of movement, such as handwritten letters, are fundamentally easier to decode than point-to-point movements. We think this is one—but not necessarily the only—important factor that enabled a handwriting BCI to go faster than continuous-motion point-and-click BCIs. Other discrete (classification-based) BCIs have also typically used directional movements with little temporal variety, which may have limited their accuracy and/or the size of the movement set[24,25].

More generally, using the principle of maximizing the nearest-neighbour distance between movements, it should be possible to optimize a set of movements for ease of classification[26]. We investigated this possibility and designed an alphabet that is theoretically easier to classify than the Latin alphabet (Extended Data Fig. 6). The optimized alphabet avoids large clusters of redundant letters that are written similarly (most Latin letters begin with either a downstroke or a counter-clockwise curl).

Discussion

Locked-in syndrome (paralysis of nearly all voluntary muscles) severely impairs or prevents communication, and is most frequently caused by brainstem stroke or late-stage amyotrophic lateral sclerosis (estimated prevalence of locked-in syndrome: 1 in 100,000[27]). Commonly used BCIs for restoring communication are either flashing electroencephalogram (EEG) spellers[18,28–32] or intracortical point-and-click BCIs[6,7,33]. EEG spellers based on oddball potentials or motor imagery typically achieve 1–5 characters per minute[28–32]. EEG spellers that use visually evoked potentials have achieved speeds of 60 characters per minute[18], but have notable usability limitations, as they tie up the eyes, are not typically self-paced and require panels of flashing lights on a screen. Intracortical BCIs based on 2D cursor movements give the user more freedom to look around and set their own pace of communication, but have yet to exceed 40 correct characters per minute in humans[7]. Recently, speech-decoding BCIs have shown exciting promise for restoring rapid communication[16,17,34], but their accuracies and vocabulary sizes require further improvement to support general-purpose use.

Here, we introduced a new approach for communication BCIs—decoding a rapid, dexterous motor behaviour in an individual with tetraplegia—that sets a benchmark for communication rate at 90 characters per minute. The real-time system is general (the user can express any sentence), easy to use (entirely self-paced and the eyes are free to move) and accurate enough to be useful in the real-world (94.1% raw accuracy, and greater than 99% accuracy offline with a large-vocabulary language model). To achieve high performance, we developed decoding methods to work effectively with unlabelled neural sequences in data-limited regimes. These methods could be applied more generally to any sequential behaviour that cannot be observed directly (for example, decoding speech from someone who can no longer speak).

It is important to recognize that the current system is a proof of concept that a high-performance handwriting BCI is possible (in a single participant); it is not yet a complete, clinically viable system. More work is needed to demonstrate high performance in additional people, expand the character set (for example, capital letters), enable text editing and deletion, and maintain robustness to changes in neural activity without interrupting the user for decoder retraining. More broadly, intracortical microelectrode array technology is still maturing, and requires further demonstrations of longevity, safety and efficacy before widespread clinical adoption[35,36]. Variability in performance across participants is also a potential concern (in a previous study, T5 achieved the highest performance of three participants[7]).

Nevertheless, we believe that the future of intracortical BCIs is bright. Current microelectrode array technology has been shown to retain functionality for more than 1,000 days after implant[37,38] (including here; see Extended Data Fig. 7), and has enabled the highest BCI performance so far compared to other recording technologies (for example, EEG or electrocorticography) for restoring communication[7], arm control[2,5]

and general-purpose computer use[39]. New developments are under way for implant designs that increase the electrode count by at least an order of magnitude, which will further improve performance and longevity[35,36,40,41]. Finally, we envision that a combination of algorithmic innovations[42–44] and improvements to device stability will continue to reduce the need for daily decoder retraining. Here, offline analyses showed the potential promise of more limited, or even unsupervised, decoder retraining (Fig. 3).

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41586-021-03506-2.

1. Hochberg, L. R. et al. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. *Nature* **485**, 372–375 (2012).
2. Collinger, J. L. et al. High-performance neuroprosthetic control by an individual with tetraplegia. *Lancet* **381**, 557–564 (2013).
3. Aflalo, T. et al. Neurophysiology. Decoding motor imagery from the posterior parietal cortex of a tetraplegic human. *Science* **348**, 906–910 (2015).
4. Bouton, C. E. et al. Restoring cortical control of functional movement in a human with quadriplegia. *Nature* **533**, 247–250 (2016).
5. 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).
6. Jarosiewicz, B. et al. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain–computer interface. *Sci. Transl. Med.* **7**, 313ra179 (2015).
7. Pandarinath, C. et al. High performance communication by people with paralysis using an intracortical brain–computer interface. *eLife* **6**, e18554 (2017).
8. Palin, K., Feit, A. M., Kim, S., Kristensson, P. O. & Oulasvirta, A. How do people type on mobile devices? Observations from a study with 37,000 volunteers. In *Proc. 21st International Conference on Human–Computer Interaction with Mobile Devices and Services* 1–12 (Association for Computing Machinery, 2019).
9. Yousry, T. A. et al. Localization of the motor hand area to a knob on the precentral gyrus. A new landmark. *Brain* **120**, 141–157 (1997).
10. Willett, F. R. et al. Hand knob area of premotor cortex represents the whole body in a compositional way. *Cell* **181**, 396–409 (2020).
11. Williams, A. H. et al. Discovering precise temporal patterns in large-scale neural recordings through robust and interpretable time warping. *Neuron* **105**, 246–259 (2020).
12. Hinton, G. et al. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal Process. Mag.* **29**, 82–97 (2012).
13. Graves, A., Mohamed, A. & Hinton, G. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* 6645–6649 (2013).
14. Xiong, W. et al. The Microsoft 2017 Conversational Speech Recognition System. Preprint at https://arxiv.org/abs/1708.06073 (2017).
15. He, Y. et al. Streaming end-to-end speech recognition for mobile devices. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing* 6381–6385 (2019).
16. Anumanchipalli, G. K., Chartier, J. & Chang, E. F. Speech synthesis from neural decoding of spoken sentences. *Nature* **568**, 493–498 (2019).
17. Makin, J. G., Moses, D. A. & Chang, E. F. Machine translation of cortical activity to text with an encoder-decoder framework. *Nat. Neurosci.* **23**, 575–582 (2020).
18. Chen, X. et al. High-speed spelling with a noninvasive brain–computer interface. *Proc. Natl Acad. Sci. USA* **112**, E6058–E6067 (2015).
19. Dickey, A. S., Suminski, A., Amit, Y. & Hatsopoulos, N. G. Single-unit stability using chronically implanted multielectrode arrays. *J. Neurophysiol.* **102**, 1331–1339 (2009).
20. Eleryan, A. et al. Tracking single units in chronic, large scale, neural recordings for brain machine interface applications. *Front. Neuroeng.* **7**, 23 (2014).
21. Downey, J. E., Schwed, N., Chase, S. M., Schwartz, A. B. & Collinger, J. L. Intracortical recording stability in human brain–computer interface users. *J. Neural Eng.* **15**, 046016 (2018).
22. Willett, F. R. et al. Signal-independent noise in intracortical brain–computer interfaces causes movement time properties inconsistent with Fitts' law. *J. Neural Eng.* **14**, 026010 (2017).
23. Gao, P. et al. A theory of multineuronal dimensionality, dynamics and measurement. Preprint at https://doi.org/10.1101/214262 (2017).
24. Musallam, S., Corneil, B. D., Greger, B., Scherberger, H. & Andersen, R. A. Cognitive control signals for neural prosthetics. *Science* **305**, 258–262 (2004).
25. Santhanam, G., Ryu, S. I., Yu, B. M., Afshar, A. & Shenoy, K. V. A high-performance brain–computer interface. *Nature* **442**, 195–198 (2006).
26. Cunningham, J. P., Yu, B. M., Gilja, V., Ryu, S. I. & Shenoy, K. V. Toward optimal target placement for neural prosthetic devices. *J. Neurophysiol.* **100**, 3445–3457 (2008).
27. Pels, E. G. M., Aarnoutse, E. J., Ramsey, N. F. & Vansteensel, M. J. Estimated prevalence of the target population for brain–computer interface neurotechnology in the Netherlands. *Neurorehabil. Neural Repair* **31**, 677–685 (2017).
28. Vansteensel, M. J. et al. Fully implanted brain–computer interface in a locked-in patient with ALS. *N. Engl. J. Med.* **375**, 2060–2066 (2016).
29. Nijboer, F. et al. A P300-based brain–computer interface for people with amyotrophic lateral sclerosis. *Clin. Neurophysiol.* **119**, 1909–1916 (2008).
30. Townsend, G. et al. A novel P300-based brain–computer interface stimulus presentation paradigm: moving beyond rows and columns. *Clin. Neurophysiol.* **121**, 1109–1120 (2010).
31. McCane, L. M. et al. P300-based brain–computer interface (BCI) event-related potentials (ERPs): people with amyotrophic lateral sclerosis (ALS) vs. age-matched controls. *Clin. Neurophysiol.* **126**, 2124–2131 (2015).
32. Wolpaw, J. R. et al. Independent home use of a brain–computer interface by people with amyotrophic lateral sclerosis. *Neurology* **91**, e258–e267 (2018).
33. Bacher, D. et al. Neural point-and-click communication by a person with incomplete locked-in syndrome. *Neurorehabil. Neural Repair* **29**, 462–471 (2015).
34. Mugler, E. M. et al. Direct classification of all American English phonemes using signals from functional speech motor cortex. *J. Neural Eng.* **11**, 035015 (2014).
35. Nurmikko, A. Challenges for large-scale cortical interfaces. *Neuron* **108**, 259–269 (2020).
36. Vázquez-Guardado, A., Yang, Y., Bandodkar, A. J. & Rogers, J. A. Recent advances in neurotechnologies with broad potential for neuroscience research. *Nat. Neurosci.* **23**, 1522–1536 (2020).
37. 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).
38. Bullard, A. J., Hutchison, B. C., Lee, J., Chestek, C. A. & Patil, P. G. Estimating risk for future intracranial, fully implanted, modular neuroprosthetic systems: a systematic review of hardware complications in clinical deep brain stimulation and experimental human intracortical arrays. *Neuromodulation* **23**, 411–426 (2020).
39. Nuyujukian, P. et al. Cortical control of a tablet computer by people with paralysis. *PLoS One* **13**, e0204566 (2018).
40. Musk, E. An integrated brain–machine interface platform with thousands of channels. *J. Med. Internet Res.* **21**, e16194 (2019).
41. Sahasrabuddhe, K. et al. The Argo: a high channel count recording system for neural recording in vivo. *J. Neural Eng.* **18**, 015002 (2021).
42. 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).
43. Dyer, E. L. et al. A cryptography-based approach for movement decoding. *Nat. Biomed. Eng.* **1**, 967–976 (2017).
44. Degenhart, A. D. et al. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. *Nat. Biomed. Eng.* **4**, 672–685 (2020).

Reporting summary

Further information on research design is available in the Nature Research Reporting Summary linked to this paper.

Data availability

All neural data needed to reproduce the findings in this study are publicly available at the Dryad repository (https://doi.org/10.5061/dryad.wh70rxwmv). The dataset contains neural activity recorded during the attempted handwriting of 1,000 sentences (43,501 characters) over 10.7 hours.

Code availability

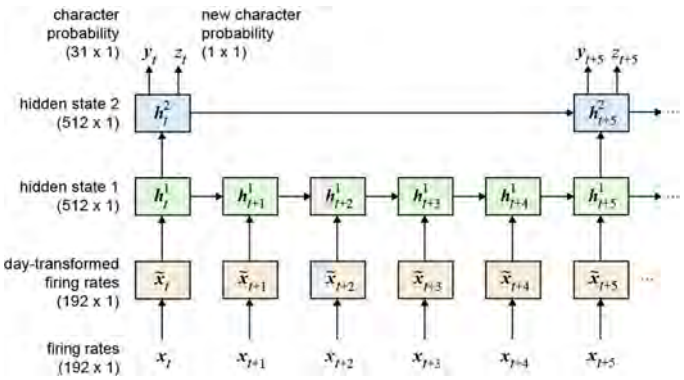
Code that implements an offline reproduction of the central findings in this study (high-performance neural decoding with an RNN) is publicly available on GitHub at https://github.com/fwillett/handwritingBCI.

Acknowledgements We thank participant T5 and his caregivers for their dedicated contributions to this research, N. Lam, E. Siauciunas and B. Davis for administrative support and E. Woodrum for the drawings in Figs. 1a, 2a. F.R.W. and D.T.A. acknowledge the support of the Howard Hughes Medical Institute. L.R.H. acknowledges the support of the Office of Research and Development, Rehabilitation R&D Service, US Department of Veterans Affairs (A2295R, N2864C); the National Institute of Neurological Disorders and Stroke and BRAIN Initiative (UH2NS095548); and the National Institute on Deafness and Other Communication Disorders (R01-DC009899, U01-DC017844). K.V.S. and J.M.H. acknowledge the support of the National Institute on Deafness and Other Communication Disorders (R01-DC014034, U01-DC017844); the National Institute of Neurological Disorders and Stroke (UH2-NS095548, U01-NS098968); L. and P. Garlick; S. and B. Reeves; and the Wu Tsai Neurosciences Institute at Stanford. K.V.S. acknowledges the support of the Simons Foundation Collaboration on the Global Brain 543045 and the Howard Hughes Medical Institute (K.V.S. is a Howard Hughes Medical Institute Investigator). The funders had no role in study design, data collection and interpretation, or the decision to submit the work for publication.

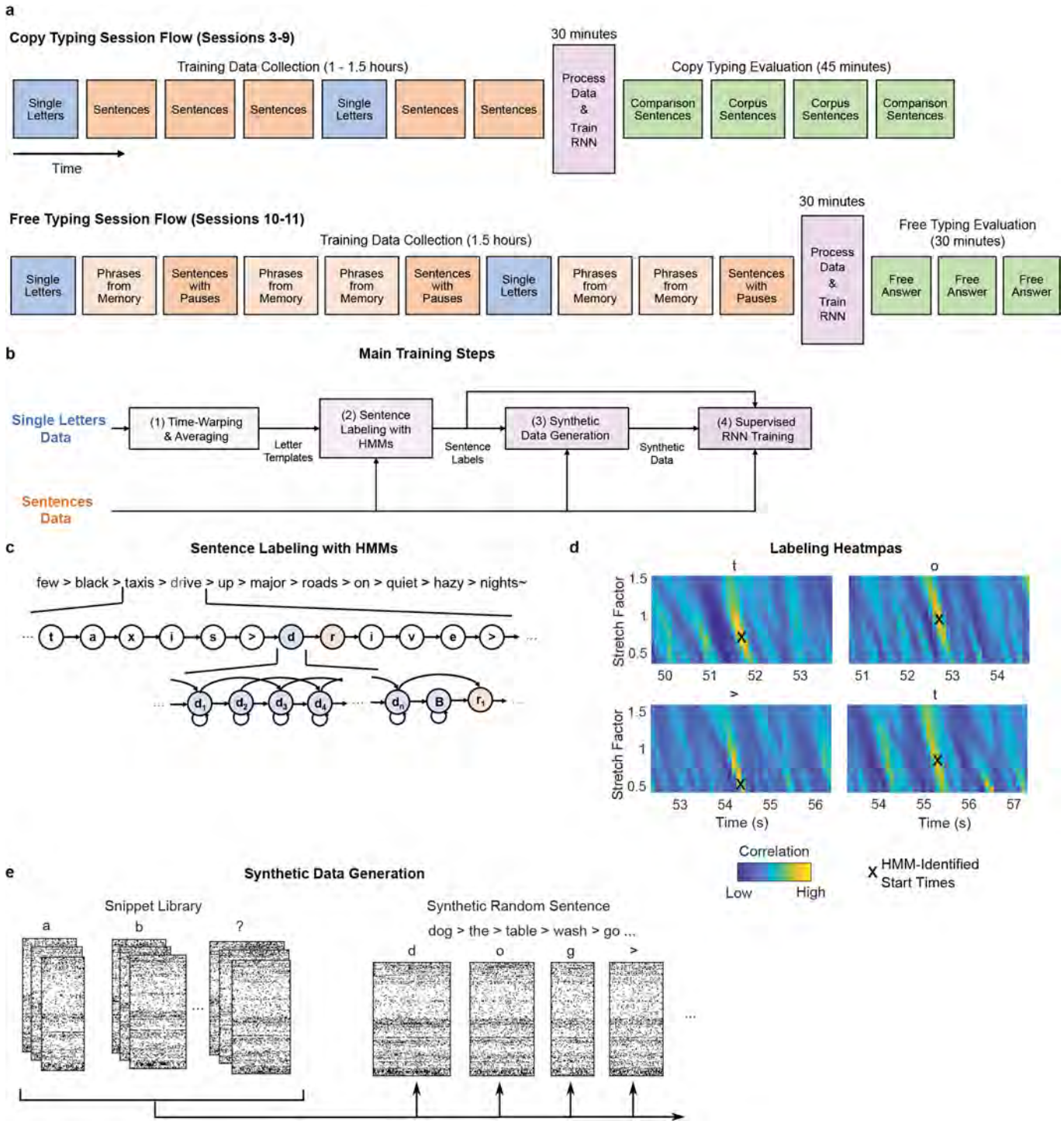
Author contributions F.R.W. conceived the study, built the real-time decoder, analysed the data and wrote the manuscript. F.R.W. and D.T.A. collected the data. L.R.H. is the sponsor-investigator of the multi-site clinical trial. J.M.H. planned and performed T5's array placement surgery and was responsible for all clinical-trial-related activities at Stanford. J.M.H. and K.V.S. supervised and guided the study. All authors reviewed and edited the manuscript.

Competing interests The MGH Translational Research Center has a clinical research support agreement with Neuralink, Paradromics and Synchron, for which L.R.H. provides consultative input. J.M.H. is a consultant for Neuralink, and serves on the Medical Advisory Board of Enspire DBS. K.V.S. consults for Neuralink and CTRL-Labs (part of Facebook Reality Labs) and is on the scientific advisory boards of MIND-X, Inscopix and Heal. F.R.W., J.M.H. and K.V.S. are inventors on patent application US 2021/0064135 A1 (the applicant is Stanford University), which covers the neural decoding approach taken in this work. All other authors have no competing interests.

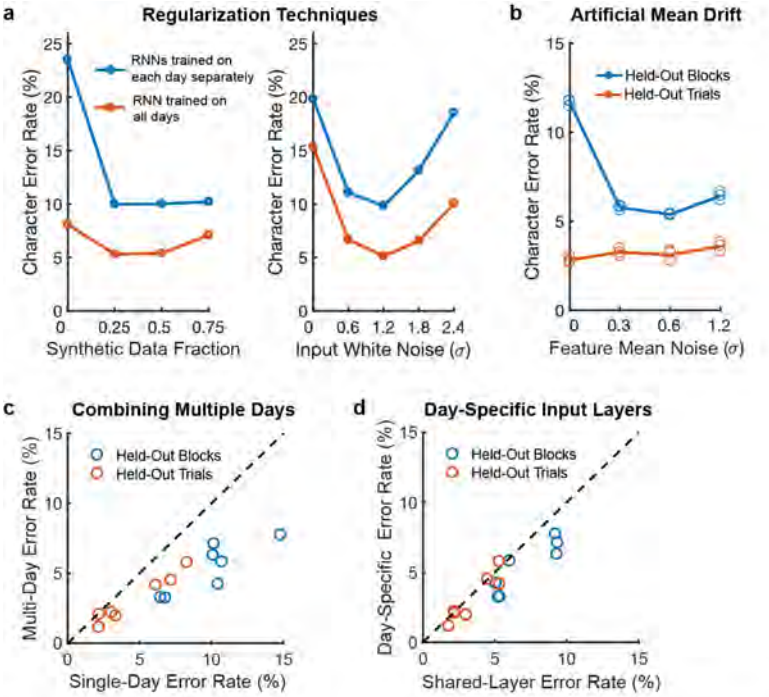
Additional information
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41586-021-03506-2.
Correspondence and requests for materials should be addressed to F.R.W.
Peer review information *Nature* thanks Karim Oweiss and the other, anonymous, reviewer(s) for their contribution to the peer review of this work. Peer reviewer reports are available.
Reprints and permissions information is available at http://www.nature.com/reprints.



Extended Data Fig. 1 | Diagram of the RNN architecture. We used a two-layer gated recurrent unit (GRU) recurrent neural network architecture to convert sequences of neural firing rate vectors $\mathbf{x}_t$ (which were temporally smoothed and binned at 20 ms) into sequences of character probability vectors $\mathbf{y}_t$ and 'new character' probability scalars z_t. The $\mathbf{y}_t$ vectors describe the probability of each character being written at that moment in time, and the z_t scalars go high whenever the RNN detects that T5 is beginning to write any new character. Note that the top RNN layer runs at a slower frequency than the bottom layer, which we found improved the speed of training by making it easier to hold information in memory for long time periods. Thus, the RNN outputs are updated only once every 100 ms. Also, note that we used a day-specific affine transform to account for day-to-day changes in the neural activity (bottom row)—this helps the RNN to account for changes in neural tuning caused by electrode array micromotion or brain plasticity when training data are combined across multiple days.



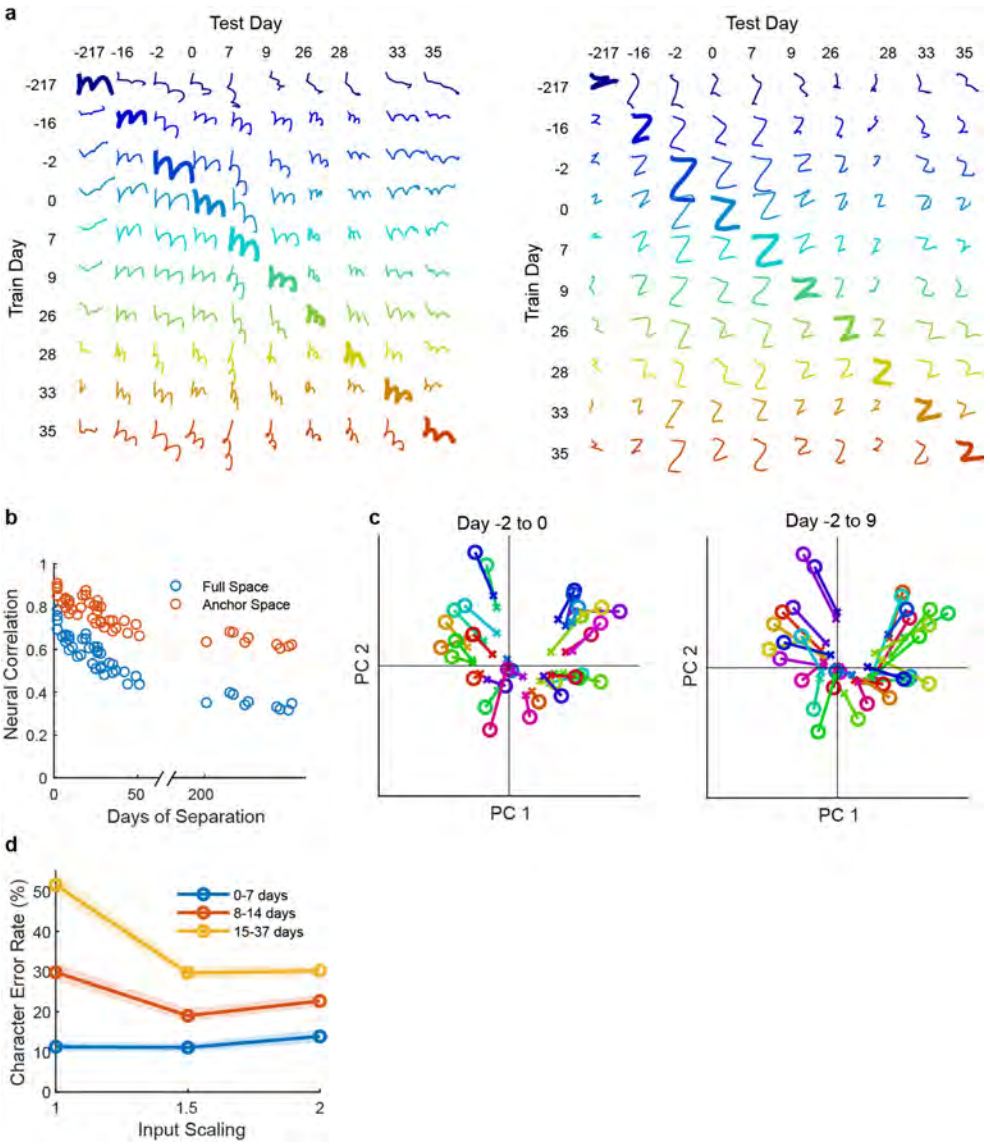
Extended Data Fig. 2 | Overview of RNN training methods. a, Diagram of the session flow for copy-typing and free-typing sessions (each rectangle corresponds to one block of data). First, single-letter and sentences training data are collected (blue and red blocks). Next, the RNN is trained using the newly collected data plus all of the previous days' data (purple block). Finally, the RNN is held fixed and evaluated (green blocks). **b**, Diagram of the data processing and RNN training process (purple block in **a**). First, the single-letter data are time-warped and averaged to create spatiotemporal templates of neural activity for each character. These templates are used to initialize the hidden Markov models (HMMs) for sentence labelling. After labelling, the observed data are cut apart and rearranged into new sequences of characters to make synthetic sentences. Finally, the synthetic sentences are combined with the real sentences to train the RNN. **c**, Diagram of a forced-alignment HMM used to label the sentence 'few black taxis drive up major roads on quiet hazy nights'. The HMM states correspond to the sequence of characters in the sentence. **d**, The label quality can be verified with cross-correlation heat maps made by correlating the single character neural templates with the real data. The HMM-identified character start times form clear hotspots on the heat maps. Note that these heat maps are depicted only to qualitatively show label quality and aren't used for training (only the character start times are needed to generate the targets for RNN training). **e**, To generate new synthetic sentences, the neural data corresponding to each labelled character in the real data are cut out of the data stream and put into a snippet library. These snippets are then pulled from the library at random, stretched or compressed in time by up to 30% (to add more artificial timing variability) and combined into new sentences.



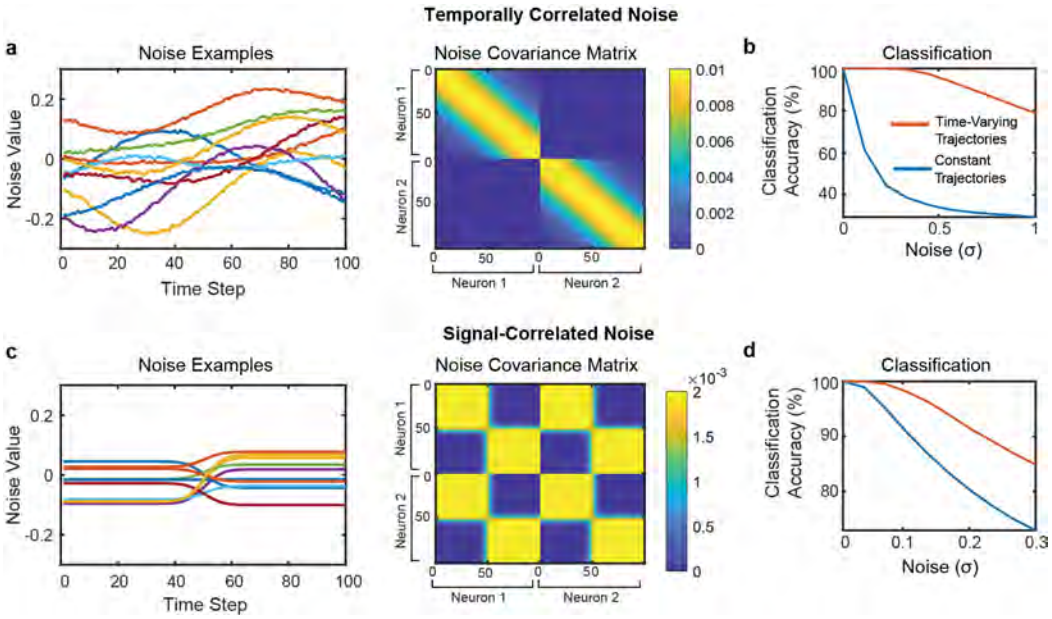
e

Parameter	Panel	Comparison	Error Rate Improvement [95% CI]
Synthetic data fraction	a (left)	Single-day RNN (0 vs. 0.25)	12.9 [11.9, 14.0]
		Multi-day RNN (0 vs. 0.25)	2.7 [2.2, 3.2]
Input white noise	a (right)	Single-day RNN (0 vs. 1.2)	9.8 [8.7, 11.0]
		Multi-day RNN (0 vs. 1.2)	10.1 [9.2, 11.1]
Feature mean noise	b	Held-out blocks (0 vs. 0.3)	5.7 [4.9, 6.5]
		Held-out trials (0 vs. 0.3)	-0.7 [-1.3, -0.1]
Single-day vs. multi-day	c	Held-out blocks	4.7 [4.1, 5.3]
		Held-out trials	1.5 [0.6, 2.4]
Shared-layer vs. day-specific layers	d	Held-out blocks	1.6 [1.1, 2.7]
		Held-out trials	0.1 [-0.7, 0.7]

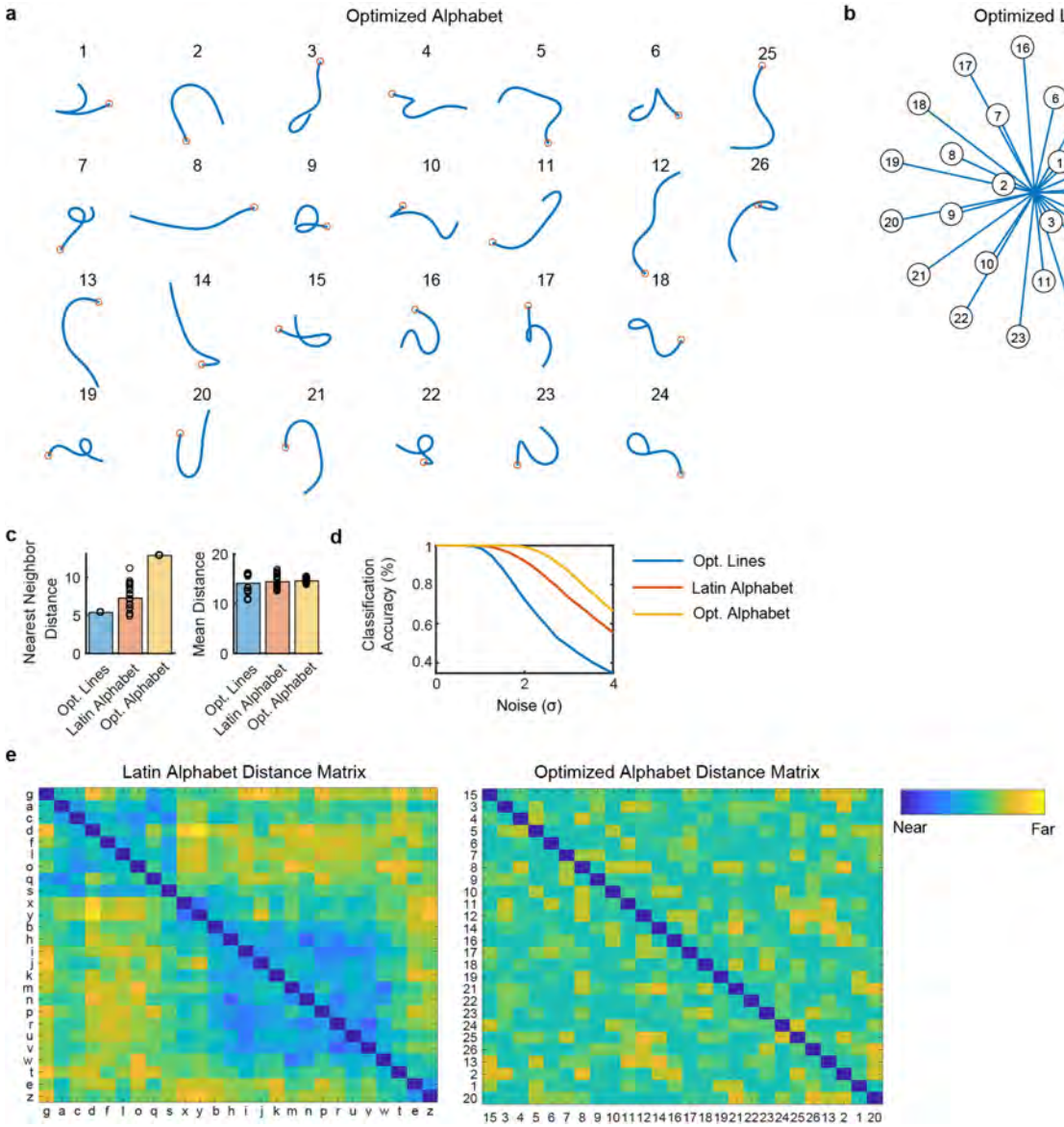
Extended Data Fig. 3 | The effect of key RNN parameters on performance. **a**, Training with synthetic data (left) and artificial white noise added to the inputs (right) were both essential for high performance. Data are shown from a grid search over both parameters, and lines show performance at the best value for the other parameter. Results indicate that both parameters are needed for high performance, even when the other is at the best value. Using synthetic data is more important when the size of the dataset is highly limited, as is the case when training on only a single day of data (blue line). Note that the inputs given to the RNN were *z*-scored, so the input white noise is in units of standard deviations of the input features. **b**, Artificial noise added to the feature means (random offsets and slow changes in the baseline firing rate) greatly improves the ability of the RNN to generalize to new blocks of data that occur later in the session, but does not help the RNN to generalize to new trials within blocks of data on which it was already trained. This is because feature means change slowly over time. For each parameter setting, three separate RNNs were trained (circles); results show low variability across RNN training runs. **c**, Training an RNN with all of the datasets combined improves performance relative to training on each day separately. Each circle shows the performance on one of seven days. **d**, Using separate input layers for each day is better than using a single layer across all days. **e**, Improvements in character error rates are summarized for each parameter. 95% CIs were computed with bootstrap resampling of single trials ($n = 10{,}000$). As shown in the table, all parameters show a statistically significant improvement for at least one condition (CIs do not intersect zero).



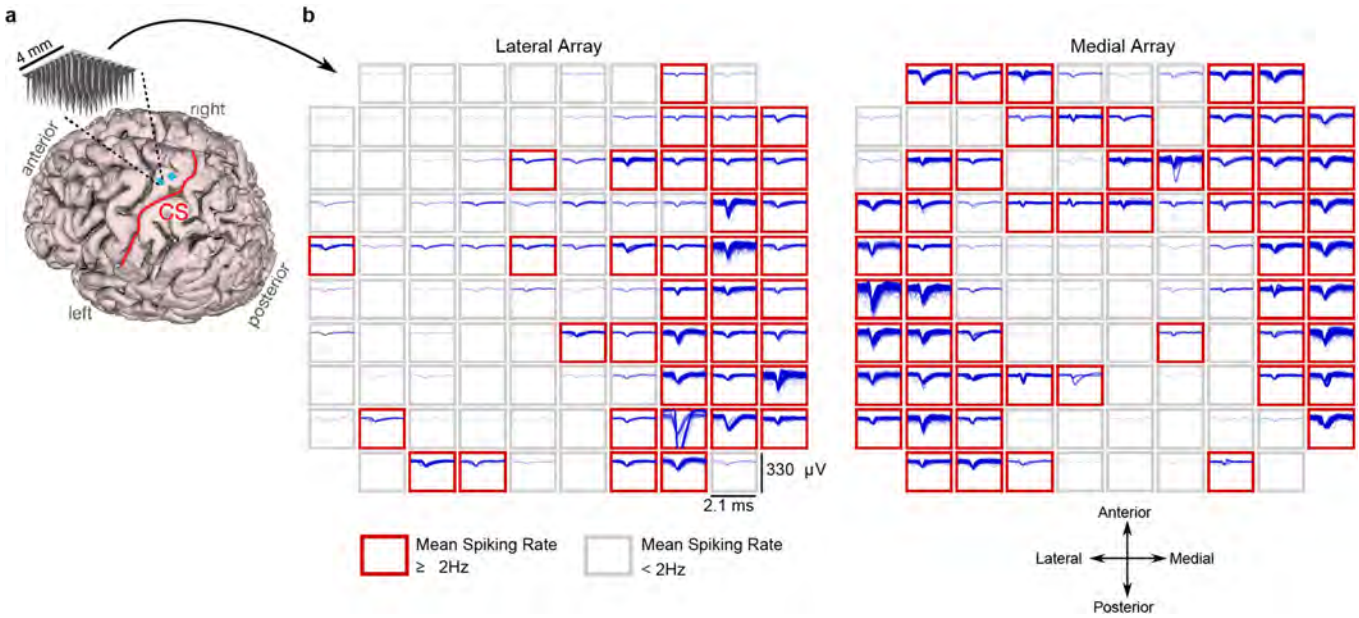
Extended Data Fig. 4 | Changes in neural recordings across days. a, To visualize how much the neural recordings changed across time, decoded pen-tip trajectories were plotted for two example letters (m and z) for all 10 days of data (columns), using decoders trained on all other days (rows). Each session is labelled according to the number of days passed relative to 9 December 2019 (day 4). Results show that although patterns of neural activity clearly change over time, their essential structure is largely conserved (as decoders trained on past days transfer readily to future days). **b**, The correlation (Pearson's *r*) between the neural activity patterns of each session was computed for each pair of sessions and plotted as a function of the number of days separating each pair. Blue circles show the correlation computed in the full neural space (all 192 electrodes), whereas red circles show the correlation in the 'anchor' space (top 10 principal components of the earlier session). High values indicate a high similarity in how characters are neurally encoded across days. The fact that correlations are higher in the anchor space suggests that the structure of the neural patterns stays largely the same as it slowly rotates into a new space, causing shrinkage in the original space but little change in structure. **c**, A visualization of how each character's neural representation changes over time, as viewed through the top two PCs of the original 'anchor' space. Each circle represents the neural activity pattern for a single character, and each x symbol shows that same character on a later day (lines connect matching characters). Left, a pair of sessions with only two days between them (day −2 to 0); right, a pair of sessions with 11 days between them (day −2 to 9). The relative positioning of the neural patterns remains similar across days, but most conditions shrink noticeably towards the origin. This is consistent with the neural representations slowly rotating out of the original space into a new space, and suggests that scaling-up the input features may help a decoder to transfer more accurately to a future session (by counteracting this shrinkage effect). **d**, Similar to Fig. 3b, copy-typing data from eight sessions were used to assess offline whether scaling-up the decoder inputs improves performance when evaluating the decoder on a future session (when no decoder retraining is used). All session pairs (X, Y) were considered. Decoders were first initialized using all data from session X and earlier, then evaluated on session Y under different input-scaling factors (for example, an input scale of 1.5 means that input features were scaled up by 50%). Lines indicate the mean raw character error rate and shaded regions show 95% CIs. Results show that when long periods of time pass between sessions, input scaling improves performance. We therefore used an input-scaling factor of 1.5 when assessing decoder performance in the 'no retraining' conditions of Fig. 3.



Extended Data Fig. 5 | Effect of correlated noise on the toy model of temporal dimensionality. a, Example noise vectors and covariance matrix for temporally correlated noise. On the left, example noise vectors are plotted (each line depicts a single example). Noise vectors are shown for all 100 time steps of neuron 1. On the right, the covariance matrix used to generate temporally correlated noise is plotted (dimensions = 200 × 200). The first 100 time steps describe the noise of neuron 1 and the last 100 time steps describe the noise of neuron 2. The diagonal band creates noise that is temporally correlated within each simulated neuron (but the two neurons are uncorrelated with each other). **b**, Classification accuracy when using a maximum likelihood classifier to classify between all four possible trajectories in the presence of temporally correlated noise. Even in the presence of temporally correlated noise, the time-varying trajectories are still much easier to classify. **c**, Example noise vectors and noise covariance matrix for noise that is correlated with the signal (that is, noise that is concentrated only in spatiotemporal dimensions that span the class means). Unlike the temporally correlated noise, this covariance matrix generates spatiotemporal noise that has correlations between time steps and neurons. **d**, Classification accuracy in the presence of signal-correlated noise. Again, time-varying trajectories are easier to classify than constant trajectories. See Supplementary Note 1 for a detailed interpretation of this figure.



Extended Data Fig. 6 | An artificial alphabet optimized to maximize neural decodability. a, Using the principle of maximizing the nearest-neighbour distance, we optimized for a set of pen trajectories that are theoretically easier to classify than the Latin alphabet (using standard assumptions of linear neural tuning to pen-tip velocity). **b**, For comparison, we also optimized a set of 26 straight lines that maximize the nearest-neighbour distance. **c**, Pairwise Euclidean distances between pen-tip trajectories were computed for each set, revealing a larger nearest-neighbour distance (but not mean distance) for the optimized alphabet compared to the Latin alphabet. Each circle represents a single movement and bar heights show the mean. **d**, Simulated classification accuracy as a function of the amount of artificial noise added. Results confirm that the optimized alphabet is indeed easier to classify than the Latin alphabet, and that the Latin alphabet is much easier to classify than straight lines, even when the lines have been optimized. **e**, Distance matrices for the Latin alphabet and optimized alphabets show the pairwise Euclidean distances between the pen trajectories. The distance matrices were sorted into seven clusters using single-linkage hierarchical clustering. The distance matrix for the optimized alphabet has no apparent structure; by contrast, the Latin alphabet has two large clusters of similar letters (letters that begin with a counter-clockwise curl, and letters that begin with a downstroke).



Extended Data Fig. 7 | Example spiking activity recorded from each microelectrode array. a, Magnetic resonance imaging (MRI)-derived brain anatomy of participant T5. Microelectrode array locations (blue squares) were determined by co-registration of postoperative CT images with preoperative MRI images. **b**, Example spike waveforms detected during a 10-s time window are plotted for each electrode (data were recorded on post-implant day 1,218). Each rectangular panel corresponds to a single electrode and each blue trace is a single spike waveform (2.1-ms duration). Spiking events were detected using a −4.5 root mean square (RMS) threshold, thereby excluding almost all background activity. Electrodes with a mean threshold crossing rate of at least 2 Hz were considered to have 'spiking activity' and are outlined in red (note that this is a conservative estimate that is meant to include only spiking activity that could be from single neurons, as opposed to multiunit 'hash'). The results show that many electrodes still record large spiking waveforms that are well above the noise floor (the *y* axis of each panel spans 330 µV, whereas the background activity has an average RMS value of only 6.4 µV). On this day, 92 electrodes out of 192 had a threshold crossing rate of at least 2 Hz.

Extended Data Table 1 | Example decoded sentences for one block of copy typing

Prompt	Real-Time Output	Real-Time Output (Annotated)	Offline Language Model (Annotated)
infected adults develop a cough and their skin and ears turn blue.	infected adults dercep a cough and thhir skin and ears tunn blue	infected adults dercep a cough and thhir skin and ears tunn blue.	infected adults develop a cough and their skin and ears turn blue.
i interrupted, unable to keep silent.	i interrupted, unabee to keep silent.	i interrupted, unabee to keep silent.	i interrupted, unable to keep silent.
i dumped the tools in the hut.	i dumped the tools in the hut.	i dumped the tools in the hut.	i dumped the tools in the hut.
within thirty seconds the army had landed.	within thirty seconds the armmy had landed	within thirty seconds the armmy had landed.	within thirty seconds the army had landed.
that's when i threw up on the carpet.	that's when i threw up on the carpet.	that's when i threw up on the carpet.	that's when i threw up on the carpet.
he didn't want to rub salt into her wounds.	he didn't want to rub salt into her wounds.	he didn't want to rub salt into her wounds.	he didn't want to rub salt into her wounds.
shouting and swearing, i yelled for an epidural.	shouting and sweerin, I yeled for an epidural.	shouting and sweering, i yelled for ▪ an epidural.	shouting and swearing, i yelled for a an epidermal.
you wish to purchase something?	you wish to purchase something?	you wish to purchase somathing?	you wish to purchase something?
lowell felt like a soldier on a battlefield, stripped of ammunition.	lowel felt like a soldier on a battlefield, stripped ef ammunition.	lowell felt like a soldier on a battlefield, stripped ef ammunition.	lowell felt like a soldier on a battlefield, stripped of ammunition.
there are only one or two minor casualties.	thee ane only on or two minor cafualties.	there ane only one or two minor cafualties.	there are only one or two minor casualties.

In the rightmost columns, errors are highlighted in red (extra spaces are denoted with a red square, and omitted letters are indicated with a strikethrough). Note that our language model substitutes 'epidermal' for 'epidural', because 'epidural' is out of vocabulary. The mean typing rate for this block was 86.47 characters per minute and the character error rates were 4.18% (real-time output) and 1.22% (language model). Sentence prompts were taken from the British National Corpus according to a random selection process (see Supplementary Methods for details).

Extended Data Table 2 | Example decoded sentences for one block of free typing

Prompt	Real-Time Output	Real-Time Output (Annotated)	Offline Language Model (Annotated)
what made you first get interested in machining?	iit was the famil business	iit was the family business	it was the family business
what is the hardest part of machining?	forming the tooling	forming the tooling	forming the tooling
how much spice do you like in your food?	lots and lots	lots and lots	lots and lots
what type of music do you enjoy most?	ii like loud rock musica	ii like loud rock musica	i like loud rock music
what are some of your favorite games to play?	i like to play cards	i like to play cards	i like to play cards
what has taken you the longest to get good or decent at?	i worked for years to perfecet my photography	i worked for years to perfecet my photography	i worked for years to perfect my photography
what food do you love that a lot of people might find a little odd?	very few poope can eat smeked sailfish	very few poople can eat smeked sailfish	very few people can eat smoked salt fish
if you could start a charity, what would it be for?	i woud proovide furding to smal families	i woud proovide furding to small families	i would provide funding to small families
what advice would you give to your younger self?	be patientt it wil got better	be patientt it will got better	be patient, it will get better

In the rightmost columns, errors are highlighted in red (omitted letters are indicated with a strikethrough). Note that our language model substitutes 'salt fish' for 'sailfish', because 'sailfish' is out of vocabulary. The mean typing rate for this block was 73.8 characters per minute and the character error rates were 6.82% (real-time output) and 1.14% (language model).

nature research

Corresponding author(s): Francis R. Willett

Last updated by author(s): Mar 14, 2021

Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒	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 all covariates tested
☐	☒	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☐	☒	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☐	☒	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	The software for running the experimental tasks, recording data, and implementing the real-time decoding system was custom developed using MATLAB and Simulink Real-Time (MathWorks, Natick, MA).
Data analysis	Data was analyzed using custom MATLAB and python code. Custom code will be made publicly available on GitHub after acceptance.

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

Data will be made publicly available on Dryad upon acceptance. Relevant links and identifiers will be included in the final manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	No sample-size calculation was performed. Data were collected in a single participant to characterize the performance of a brain-computer interface. Uncertainty in performance estimates were quantified with confidence intervals, and show a robust result.
Data exclusions	This study is based on brain-computer interface performance evaluation data collected over a series of days. All days are reported in the study and all relevant data is included.
Replication	This study assessed brain-computer interface performance in a single participant. Results were replicated across multiple days of performance evaluation.
Randomization	Randomization into groups is not relevant for this study - only one participant was included in this study.
Blinding	Blinding is not relevant to this study, which assessed the performance of a brain-computer interface in a single individual.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☐	☒ Human research participants
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Human research participants

Policy information about studies involving human research participants

Population characteristics	This study includes data from one participant (identified as T5) who gave informed consent and was enrolled in the BrainGate2 Neural Interface System clinical trial (ClinicalTrials.gov Identifier: NCT00912041, registered June 3, 2009) - but note that this study does not report clinical trial results. T5 is a right-handed man, 65 years old at the time of data collection, with a C4 AIS C spinal cord injury that occurred approximately 9 years prior to study enrollment.
Recruitment	Participant T5 was enrolled in the BrainGate2 pilot clinical trial prior to the design and execution of this study, after meeting inclusion criteria based in part on disease characteristics. Inclusion and exclusion criteria are available online (ClinicalTrials.gov).
Ethics oversight	The BrainGate2 Neural Interface System clinical trial was approved under an Investigational Device Exemption (IDE) by the US Food and Drug Administration (Investigational Device Exemption #G090003). Permission was also granted by the Institutional Review Boards of Stanford University (protocol #20804). All research was performed in accordance with relevant guidelines/ regulations.

Note that full information on the approval of the study protocol must also be provided in the manuscript.

nature portfolio

https://doi.org/10.1038/s41586-021-03506-2

Supplementary information

High-performance brain-to-text communication via handwriting

In the format provided by the authors and unedited

High-performance brain-to-text communication via handwriting

Francis R. Willett, Donald T. Avansino, Leigh R. Hochberg, Jaimie M. Henderson*, Krishna V. Shenoy*

Supplementary Information

I. Methods

1. Experimental procedures

1.1 Study participant

This study includes data from one participant (identified as T5) who gave informed consent and was enrolled in the BrainGate2 Neural Interface System clinical trial (ClinicalTrials.gov Identifier: NCT00912041, registered June 3, 2009). This pilot clinical trial was approved under an Investigational Device Exemption (IDE) by the US Food and Drug Administration (Investigational Device Exemption #G090003). Permission was also granted by the Institutional Review Boards of Stanford University (protocol #20804). T5 gave consent to publish photographs and videos containing his likeness. All research was performed in accordance with relevant guidelines / regulations.

T5 is a right-handed man, 65 years old at the time of data collection, with a C4 AIS C (ASIA Impairment Scale C – Motor Incomplete) spinal cord injury that occurred approximately 9 years prior to study enrollment. In August 2016, two 96 electrode intracortical arrays (Neuroport arrays with 1.5-mm electrode length, Blackrock Microsystems, Salt Lake City, UT) were placed in the hand "knob" area of T5's left-hemisphere (dominant) precentral gyrus. Data are reported from post-implant days 994 to 1246 (see Table M1 below for a list of all sessions). Array placement locations registered to MRI-derived brain anatomy are shown in Extended Data Fig. 7. Note that both arrays still recorded high-quality spiking activity from many electrodes. On average, 81.9 ± 5.6 (mn ± sd) out of 192 electrodes recorded spike waveforms each day at a rate of at least 2 Hz when using a spike-detection threshold of -4.5 RMS, where RMS is the electrode-specific root mean square of the voltage time series recorded on that electrode (see Extended Data Fig. 7 for example waveforms).

T5 retained full movement of the head and face and the ability to shrug his shoulders. Below the injury, T5 retained some very limited voluntary motion of the arms and legs that was largely restricted to the left elbow; however, some micromotions of the right hand were visible during attempted handwriting (see Supplementary Video 2 for hand micromotions). T5's neurologic exam findings were as follows for muscle groups controlling the motion of his right hand: Wrist Flexion=0, Wrist Extension=2, Finger Flexion=0, Finger Extension=2 (MRC Scale: 0=Nothing, 1=Muscle Twitch but no Joint Movement, 2=Some Joint Movement, 3=Overcomes Gravity, 4=Overcomes Some Resistance, 5=Overcomes Full Resistance). See our prior work for full neurologic exam results for T5 (Willett et al., 2020).

In a recent study from our group which included data from participant T5, we found that body parts which T5 still had control over (e.g. head, shoulder) did *not* have a stronger neural representation than body parts which were fully or almost fully paralyzed (Willett et al., 2020); thus, T5's limited hand motion likely did not have a large effect on the neural activity, which seems to be generated primarily by the intention to move and not overt motion itself.

Interestingly, although T5's hand was almost completely still during attempted handwriting (and he did not hold a pen), T5 reported *feeling* as though an imaginary pen in his hand was physically moving and tracing out the letter shapes as he attempted to handwrite. This subjective feeling of motion appeared to obey some physical constraints, as T5 reported being able to "write" more quickly if he attempted to write smaller letters.

1.2 Neural signal processing

Neural signals were recorded from the microelectrode arrays using the NeuroPort™ system (Blackrock Microsystems). More details are described in (Hochberg et al., 2006; Jarosiewicz et al., 2015; Pandarinath et al., 2017). Neural signals were analog filtered from 0.3 Hz to 7.5 kHz and digitized at 30 kHz (250 nV resolution). Next, a common average reference filter was applied that subtracted the average signal across the array from every electrode to reduce common mode noise. Finally, a digital bandpass filter from 250 to 3000 Hz was applied to each electrode before threshold crossing detection. This filter was applied non-causally (using a 4 ms delay) in order to improve spike detection (Masse et al., 2014).

We used multiunit threshold crossing rates as neural features for analysis and neural decoding (as opposed to spike-sorted single units). We made this choice to simplify the methods, not because action potential ("spike") waveforms could not be recorded - see Extended Data Fig. 7 for example waveforms. Recent results suggest that neural population structure can be accurately estimated from threshold crossing rates alone (Trautmann et al., 2019), and that neural decoding performance is comparable (within 5%) to using sorted units (Chestek et al., 2011; Christie et al., 2014) – although see also (Todorova et al., 2014). For threshold crossing detection, we used a negative 3.5 x RMS threshold applied to each electrode, where RMS is the electrode-specific root mean square of the voltage time series recorded on that electrode. Threshold crossing times were "binned" into 10 ms bins (for analysis) or 20 ms bins (for decoding) to estimate the threshold crossing rate in each bin. For each bin, the estimated rate was equal to the number of threshold crossings in that bin divided by the bin duration.

1.3 Overview of data collection sessions

Neural data were recorded in 3-5 hour "sessions" on scheduled days, which typically occurred 2-3 times per week. During the sessions, T5 sat upright in a wheelchair with his hand resting on his lap. A computer monitor placed in front of T5 indicated which sentence (or single character) to write and when. Data were collected in a series of 5-10 minute "blocks" consisting of an uninterrupted series of trials. In between these blocks, T5 was encouraged to rest as needed. The software for running the experimental tasks, recording data, and implementing the real-time decoding system was developed using MATLAB and Simulink Real-Time (MathWorks, Natick, MA). Table M1 below is an exhaustive list of all 11 data collection sessions reported in this work.

Session Number	Date (Post-Implant Day)	Description	Data
1	2019.05.08 (994)	Sentence-writing and character-writing pilot day (no real-time decoding)	• Sentence writing collected as training data (102 sentences, no decoding) • Single character writing (27 repetitions per character)
2	2019.06.03 (1020)	Attempted handwriting of straight lines	• Instructed delay straight-line writing (24 repetitions each for 48 straight-line conditions)
3	2019.11.25 (1195)	Real-time decoding pilot day	• Sentence writing for training data (50 sentences, no decoding) • Sentence writing for performance evaluation (34 sentences, with real-time decoding) • Single character writing (10 repetitions per character)
4	2019.12.09 (1209)	Real-time decoding pilot day	
5	2019.12.11 (1211)	Copy-typing evaluation	
6	2019.12.18 (1218)	Copy-typing evaluation	
7	2019.12.20 (1220)	Copy-typing evaluation	
8	2020.01.06 (1237)	Copy-typing evaluation	
9	2020.01.08 (1239)	Copy-typing evaluation	
10	2020.01.13 (1244)	Free-answer evaluation	• Sentence writing with artificial pauses (30 sentences, no decoding) • Phrase writing from memory (100 phrases, no decoding) • Sentence writing for performance evaluation (25 free-answer questions) • Single character writing (10 repetitions per character)
11	2020.01.15 (1246)	Free-answer evaluation	

Table M1. List of all data collection sessions included in this study.

1.4 Instructed delay paradigm

All tasks employed an instructed delay paradigm. For the single character writing task shown in Fig. 1a, the delay period duration was drawn from an exponential distribution (mean of 2.5 s); values that fell outside of the range of 2.0 – 3.0 s were re-drawn. After the delay period, the text prompt changed to "Go" and the red square (stop cue) turned green for 1 second, cueing T5 to begin attempting to write.

During sentence writing blocks, the delay period always lasted 5 seconds. During this delay period, the upcoming sentence was displayed on the screen, providing T5 time to read through it and prepare to write it. After the delay period, the red stop cue then turned green, and the sentence remained displayed on the screen while T5 attempted to handwrite it letter by letter. When T5 finished writing the sentence, he turned his head to the right, which our system detected and automatically triggered the next sentence. Head position was tracked optically with the OptiTrack V120:Trio bar (Corvallis, OR) containing three infrared cameras that tracked the position of optical markers worn on a head band.

1.5 Decoder evaluation sessions

Sentence-writing days where real-time decoding was tested (sessions 3-11) had the following structure (illustrated in Extended Data Fig. 2a). First, we collected interleaved blocks of single character writing (2 blocks, 5 repetitions of each character per block) and sentence writing (5-8 blocks with 10 sentences or 20 phrases per block); no decoder was active during these blocks. Then, we retrained the decoder using these blocks of data (combined with data from all past sessions). Finally, we collected evaluation blocks where T5 used the decoder to copy sentences (sessions 3-9, 4 blocks per session) or freely answer questions (sessions 10-11, 3 blocks per session). Note that the reported data in Fig. 2 are from sessions 5-9, since sessions 3-4 were pilot sessions devoted to exploring different decoding approaches.

Since no backspace functionality was implemented and T5 had no ability to correct errors, T5 reported spending most of his time looking at the "prompt" (the sentence he was instructed to copy) instead of watching the decoded letters appear on the screen below it. Eye tracking data confirmed that T5 spent 93% of the time looking at the prompt during the copy typing task as opposed to the real-time decoder output (Tobii 4C eye tracker).

Note that since the handwriting BCI is entirely self-paced, T5 determines the speed of the BCI (characters per minute) by choosing how quickly to attempt to write each character. We instructed T5 to proceed as quickly as possible. T5 reported to us that he increased his writing speed over time as he gained confidence that the BCI could maintain its accuracy at high speeds.

1.6 Sentence selection

<u>Copy typing sessions</u>

In each copy typing session (sessions 3-11), 5 blocks of training data (with 10 sentences each) were always collected before decoder evaluation for the purposes of decoder retraining (see

Extended Data Fig. 2a for a session flow diagram). Sentences were chosen from the British National Corpus (BNC) using the Sketch Engine tool. First, we randomly selected words from a list of the top 2,000 most common words in the BNC. Then, for each randomly chosen word, the BNC was searched for example sentences containing that word. From these examples, we hand-selected sentences of reasonable length (no more than 120 characters) and whose meaning was not too confusing out of context, so as not to be distracting to T5. The end result was a diverse sample of sentences from many different contexts (spoken English, fiction, non-fiction, news, etc.). Finally, we also included 5 pangrams (sentences containing all 26 letters) in each session's training data that did not appear in the BNC, in order to increase the frequency of rare letters.

After collecting training data, the RNN decoder was retrained and then evaluated on four evaluation blocks. Two of the four evaluation blocks always used the 7 sentences employed in (Pandarinath et al., 2017) for a direct comparison to this prior state-of-the-art point-and-click typing BCI (Supplementary Video 3). The other two evaluation blocks contained 10 unique sentences selected from the BNC (according to the same selection process described above).

Importantly, the RNN decoder was never evaluated on a sentence that it had been trained on, and every sentence was unique (except for the "direct comparison" blocks that always used the same 7 sentences from (Pandarinath et al., 2017)). When we retrained the decoder each day before performance evaluation, we retrained it using all previously collected data (from all prior days) *except* for these direct comparison blocks, in order to prevent the RNN from overfitting to these repeated sentences.

Free typing sessions

The two free-typing sessions (sessions 10-11) used 8 blocks of sentence-writing data for decoder training (instead of 5) and used a different set of sentences to add more realistic variability to the training data. For 3 of the 8 sentence-writing blocks, we randomly added hash mark characters (#) to sentences selected from the BNC, which signaled T5 to take a short, artificial pause from writing at different points in the sentence. For the other 5 blocks, we used 2-4 word phrases instead of complete sentences from the BNC, and asked T5 to write them from memory (instead of copying what was on the screen). To enforce writing from memory, we removed the phrase from the screen during the "Go" period. These features were designed to train the RNN to be robust to irregular writing speeds and unpredictable pauses that might occur more often during free typing.

2. Neural representation of attempted handwriting (Fig. 1)

2.1 PCA visualization and time-warping

To create the data visualization shown in Fig. 1b-c, threshold crossing rates were first binned into 10 ms bins and smoothed by convolving with a Gaussian kernel (sd = 30 ms) to remove high frequency noise. To find the top 3 PCs used to visualize the neural activity, the smoothed rates were then compiled into a matrix of dimension N x TC, where N is the number of microelectrodes (192), T is the number of 10 ms time bins (200), and C is the number of characters (31). Each row contains the trial-averaged response of a single electrode to each character, in a time window from -500 ms to 1500 ms around the go cue (the 31 trial-averaged responses were concatenated together into a single vector). Principal components analysis was then applied to the columns of this matrix to find the top 3 PCs used for data visualization.

Next, we used time-warped PCA (https://github.com/ganguli-lab/twpca) (Poole et al., 2017; Williams et al., 2020) to find continuous, regularized time-warping functions that align all trials belonging to the same character together (Fig. 1c). We verified that these warping functions appeared close to the identity line, smoothly bending away from it after the go cue in order to account for variations in writing speed from trial to trial (as can be seen in the example shown in Fig. 1b-c). We used the following time-warping parameters: 5 components, 0.001 scale warping regularization (L1), and 1.0 scale time regularization (L2).

2.2 Pen trajectory visualization

Overview

To construct the character traces shown in Fig. 1d, we trained a linear decoder to readout pen tip velocity from the neural activity as follows:

$$\boldsymbol{v}_t = \boldsymbol{D}\boldsymbol{x}_t + \boldsymbol{b}$$

Here, $\boldsymbol{v}_t$ is a 2 x 1 pen tip velocity vector containing X and Y velocity at time t, $\boldsymbol{D}$ is a 2 x 192 decoding matrix, $\boldsymbol{x}_t$ is a 192 x 1 vector of binned threshold crossing rates, and $\boldsymbol{b}$ is a 2 x 1 offset term. Importantly, decoding was cross-validated by holding out each character in a leave-one-out fashion. That is, the pen tip velocities for any given character were obtained using a decoder that was trained on all *other* characters, preventing the decoder from trivially overfitting to high-dimensional neural data.

To train the decoder, we used hand-made templates that describe each character's pen trajectory. The character templates were made by drawing each character with a computer mouse in the same way as T5 described writing the character. As each character was drawn, the X and Y velocity trajectories of the mouse pointer were recorded. These templates then defined the target velocity vector for the decoder on each time step of each trial, much like prior work has trained decoders to predict the user's "intended" velocity for continuous movement tasks (Gilja et al., 2012; Collinger et al., 2013; Gilja et al., 2015). These templates were only intended to be a rough approximation of T5's intended pen tip velocities, based on the assumption that

another person drawing the same character shape with a computer mouse would naturally follow a similar velocity trajectory (up to some time-scaling factor, to account for differences in overall writing speed). Nevertheless, the reconstructed pen tip velocities that were decoded in Fig. 1d were well correlated with the mouse templates (r = 0.74 across all characters).

Since drawing the characters with a computer mouse may not yield the same overall writing speed and reaction times as T5, the reaction time and time-scaling factor for each velocity template must also be optimized along with the decoder. We therefore trained the decoder in an iterative process as follows:

(1) Train the linear decoder using the currently-identified reaction time and time-scaling factors, using ordinary least squares regression to minimize the error between the template velocities and the decoded velocities.

(2) Apply the newly-trained decoder to the neural data to generate decoded velocities.

(3) Optimize the reaction times (template start times) and time-scaling factors (linear time stretching/shrinking) via a grid search to best match the current decoded velocities.

(4) Return to Step 1 until 3 iterations have been performed.

Finally, to visualize the pen trajectories for each character, the decoder was applied to time-warped and trial-averaged neural activity. Trial-averaging denoises the decoder output, and time-warping aligns all the trials together in time so that they can be averaged without losing detail. The decoded velocity was then integrated (cumulatively summed) to yield a pen tip position trajectory.

Implementation details

Each trial of neural activity was represented by a matrix of binned and smoothed threshold crossing rates with dimensions 200 x 192 (200 time steps and 192 electrodes) taken from a -0.5 to 1.5 second window around the go cue. Then, these matrices were concatenated vertically to form a predictor matrix for the linear regression of size 200*N x 192, where N is the total number of trials (N=864). Finally, a column of ones was added to the predictor matrix to create a constant offset term, yielding a design matrix $\boldsymbol{X}$ of dimension 200*N x 193. We also constructed a response matrix $\boldsymbol{Y}$ of dimension 200*N x 2 containing the template velocity vectors at each time step that the decoder should predict.

For the first iteration, $\boldsymbol{Y}$ was constructed by setting the entries of each trial equal to the velocity vectors in that character's template. The decoder coefficient matrix $\boldsymbol{B}$ was then computed with ordinary least squares to minimize the following cost function:

$$\|\boldsymbol{Y} - \boldsymbol{XB}\|_F^2$$

Here, *F* denotes the Frobenius norm (i.e., the square root of the sum of squared entries in the matrix $\boldsymbol{Y} - \boldsymbol{XB}$). For the subsequent two iterations, we used the decoded velocity vectors $\boldsymbol{XB}$ to

time-scale and shift the character templates before constructing ***Y***. We performed a grid search for each character, searching for possible template shift-times between -0.4 and 0.4 seconds and possible template time-scaling factors between 0.5 and 2.0. Templates were time-scaled by resampling with linear interpolation to stretch/shrink them in time. The shift-time and time-scale factors that maximized the correlation (Pearson's *r*) between the template and the previously decoded velocity vectors were then used to construct ***Y***.

2.3 Fraction of variance accounted for by pen velocity

To estimate how much of the neural activity can be explained by pen tip velocity (30%), we computed the fraction of variance accounted for in the *trial-averaged* (and time-warped) neural activity by a linear encoding model that describes neural activity as a linear function of the decoded pen tip velocity trajectories.

To denoise the result, we only analyzed neural activity in the top 10 neural dimensions found by principal components analysis (see above section 2.1 for PCA details). Otherwise, noise in higher dimensions can dominate the result.

The encoding model was fit with ordinary least squares regression and can be described as follows:

$$\boldsymbol{f}_t = \boldsymbol{E}\boldsymbol{v}_t + \boldsymbol{b}$$

Here, $\boldsymbol{f}_t$ is a 10 x 1 vector of trial-averaged neural activity in the top 10 neural dimensions, $\boldsymbol{E}$ is a 10 x 2 encoding matrix (of preferred directions), $\boldsymbol{v}_t$ is a 2 x 1 pen tip velocity vector containing reconstructed X and Y velocity at time *t* (and was constructed by decoding the pen tip velocity as described above in section 2.2), and $\boldsymbol{b}$ is a 10 x 1 offset term. We included all neural activity within a time window from 100 ms after the "Go" cue up until when each letter was finished being written (as determined by visual inspection of the pen tip trajectories).

The fraction of variance accounted for (FVAF) by the pen tip velocity was then computed as follows:

$$FVAF = 1 - \frac{SS_{err}}{SS_{tot}} = 1 - \frac{\sum_{t=1}^{T}(\boldsymbol{E}\boldsymbol{v}_t - \boldsymbol{f}_t)^T(\boldsymbol{E}\boldsymbol{v}_t - \boldsymbol{f}_t)}{\sum_{t=1}^{T}\boldsymbol{f}_t^T\boldsymbol{f}_t}$$

Here, *T* is the total number of time steps, SS_{err} is the sum of squared errors, and SS_{tot} is the total variance.

Because only 2D pen tip velocity was modeled (i.e., lifting-off-the-page motion was not considered here), we only included characters that required no pen-lifting in this analysis (a, b, c, d, e, g, h, l, m, n, o, p, q, r, s, u, v, w, z, >, ~).

2.4 t-SNE and k-NN classifier

For Fig. 1e, we used t-distributed stochastic neighbor embedding (t-SNE) (Maaten and Hinton, 2008) to nonlinearly reduce the dimensionality of trials of neural activity for visualization (perplexity=40). Before applying t-SNE, we smoothed the neural activity and reduced its dimensionality to 15 with PCA (using the methods described above in 2.1). Each trial of neural activity was thus represented by a 140 x 15 matrix (140 time bins by 15 dimensions, with the time window spanning 0.1 to 1.5 seconds after the go cue). We applied t-SNE to these matrices using a "time-warp" distance function that serves to account for differences in writing speed across trials, so that the same pattern of neural activity occurring at a different speed is considered nearby.

The time-warp distance function, which returns a distance between any two neural activity matrices ***X*** and ***Y***, can be written as follows:

$$d(\boldsymbol{X}, \boldsymbol{Y}) = \underset{\alpha \in [0.7, 1.42]}{\operatorname{argmin}} \frac{1}{g(\alpha)} \left\| \boldsymbol{X}_{1:g(\alpha)} - f(\boldsymbol{Y}, \alpha) \right\|_F^2$$

Here, ***X*** and ***Y*** are matrices representing single trials of neural activity, α is a time-warp factor, *f* is a warping function that returns a time-warped version of ***Y***, the subscript notation *1:g(α)* indicates to take only the time steps from 1 to *g(α)* from the matrix ***X***, and *g* is a function that returns how many time steps to consider when using the warping factor α. Essentially, this distance function returns the minimum distance over a set of possible time-warping factors.

The function *f* time-warps a trial of neural activity by a factor of *α* by resampling it in time using linear interpolation. After warping, only the first *N* shared time bins between ***X*** and ***Y*** are used to compute the distance (and the distance is normalized by dividing by the number of time bins used) – this is indicated in the equation using the function *g(α)* which returns the number of shared time bins *N* for a given warping factor. When computing the distance function, we searched over a finite set of 15 possible *α* values that were evenly spaced between 0.7 and 1.42.

We also used the time-warp distance function to perform k-nearest neighbor classification offline (k=10), which resulted in 94.1% accuracy (compared to 88.8% accuracy with a Euclidean distance function). This classification analysis was performed using leave-one-out cross-validation. The 95% confidence interval reported in the main text was a binomial proportion confidence interval (Clopper-Pearson).

3. Decoder performance metrics

3.1 Error rate and characters per minute

Character error rate was defined as the edit distance between the decoded sentence and the prompt (i.e., the number of insertions, deletions or substitutions required to make the strings of characters match exactly). Similarly, word error rate was the edit distance defined over sequences of "words" (strings of characters separated by spaces; punctuation was included as part of the word it appeared next to). For the free typing sessions, the intended sentence was determined by discussing with the participant his intended meaning.

Note that the reported error rates are the combined result of many independent sentences. To combine data across multiple sentences, we summed the number of errors across all sentences and divided this by the total number of characters/words across all sentences (as opposed to computing error rate percentages first for each sentence and then averaging the percentages). This helps prevent very short sentences from overly influencing the result.

Characters per minute was defined as 60N/(E-S), where N was the number of characters in the target sentence, E was the end time and S was the start time (in seconds). For copy typing, S was the time of the go cue and E was the time of the last decoded character. Rarely, T5 had lapses of attention and did not respond to the go cue until many seconds later; we thus capped his reaction time (the time between the go cue and the first decoded character) to no more than 2 seconds. For the free typing sessions, we defined S as the time of the first decoded character (instead of the go cue), since T5 often took substantial time after the go cue to formulate his response to the prompt.

3.2 Data exclusion

We removed a small number of trials with incomplete data when reporting decoder performance (9 out of 220 = 4%). Three copy typing trials were removed because T5 accidentally triggered the next sentence by moving his head too far to the right before he had finished typing the sentence (our system triggered the next sentence upon detection of a rightward head turn). During free typing, we removed one sentence where T5 could not think of a response and wanted to skip the question, and one sentence where we were unable to determine T5's intended spelling of a restaurant name. Finally, in one copy typing session, a software bug incorrectly initialized the RNN decoder at the beginning of each block; for this session, we excluded the first trial of each block (4 trials total).

3.3 Able-bodied smartphone typing rate

To estimate the able-bodied smartphone typing rate of people in T5's age group (115 characters per minute, as mentioned in the Summary), we used the publicly available data from (Palin et al., 2019). We took the median over all participants greater than 60 years old (T5 was 65 at the time of data collection).

4. RNN architecture

We used a two layer, gated recurrent unit RNN (Cho et al., 2014) to convert T5's neural activity into a time series of character probabilities (see Extended Data Fig. 1 for a diagram). We found that a recurrent neural network decoder outperformed a simple hidden Markov model (HMM) decoder (see section "7. Comparison to an HMM decoder" for more details).

As a pre-processing step, threshold crossing rates were binned in 20 ms time steps, z-scored (mean-subtracted and divided by the standard deviation), causally smoothed by convolving with a Gaussian kernel (sd = 40 ms) that was delayed by 100 ms, concatenated into a 192 x 1 vector $\boldsymbol{x}_t$, and then transformed using a day-specific affine layer to account for differences in neural representations across days, yielding the day-transformed vector $\tilde{\boldsymbol{x}}_t$ that was fed as input to the RNN.

The day-specific affine transform was parameterized as follows, where $\boldsymbol{A}_i$ is a 192 x 192 matrix for day *i* and $\boldsymbol{b}_i$ is a 192 x 1 bias vector for day *i*:

$$\tilde{\boldsymbol{x}}_t = \boldsymbol{A}_i \boldsymbol{x}_t + \boldsymbol{b}_i$$

For the RNN layers, we used the following variant of the gated recurrent unit RNN that is implemented by the cuDNN library (Chetlur et al., 2014):

$$\begin{aligned}
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r \tilde{\boldsymbol{x}}_t + \boldsymbol{R}_r \boldsymbol{h}_{t-1} + \boldsymbol{b}_{Wr} + \boldsymbol{b}_{Rr}) \\
\boldsymbol{u}_t &= \sigma(\boldsymbol{W}_u \tilde{\boldsymbol{x}}_t + \boldsymbol{R}_u \boldsymbol{h}_{t-1} + \boldsymbol{b}_{Wu} + \boldsymbol{b}_{Ru}) \\
\boldsymbol{c}_t &= \sigma_h(\boldsymbol{W}_h \tilde{\boldsymbol{x}}_t + \boldsymbol{r}_t * (\boldsymbol{R}_h \boldsymbol{h}_{t-1} + \boldsymbol{b}_{Rh}) + \boldsymbol{b}_{Wh}) \\
\boldsymbol{h}_t &= (1 - \boldsymbol{u}_t) * \boldsymbol{c}_t + \boldsymbol{u}_t * \boldsymbol{h}_{t-1}
\end{aligned}$$

Here, σ is the logistic sigmoid function, σ_h is the hyperbolic tangent, $\tilde{\boldsymbol{x}}_t$ is the input vector at time step *t*, $\boldsymbol{h}_t$ is the hidden state vector, $\boldsymbol{r}_t$ is the reset gate vector, $\boldsymbol{u}_t$ is the update gate vector, $\boldsymbol{c}_t$ is the candidate hidden state vector, $\boldsymbol{W}$, $\boldsymbol{R}$ and $\boldsymbol{b}$ are weight matrices and bias vectors, and * denotes element-wise multiplication. We used a two-layer RNN architecture, meaning that the hidden state of the first layer was fed as input to the second layer.

Importantly, the RNN was trained with an output delay. That is, the RNN was trained to predict the character probabilities from 1 second in the past; this was necessary to ensure that the RNN had enough time to process the entire character before deciding on its identity. The output probabilities were computed from the hidden state of the second layer as follows:

$$\begin{aligned}
\boldsymbol{y}_t &= \mathrm{softmax}(\boldsymbol{W}_y \boldsymbol{h}_t^2 + \boldsymbol{b}_y) \\
z_t &= \sigma(\boldsymbol{W}_z \boldsymbol{h}_t^2 + \boldsymbol{b}_z)
\end{aligned}$$

Here, σ is the logistic sigmoid function, $\boldsymbol{h}_t^2$ is the hidden state of the second layer, $\boldsymbol{W}$ and $\boldsymbol{b}$ are weight matrices and bias vectors, $\boldsymbol{y}_t$ is a vector of character probabilities (one entry for each character), and z_t is a scalar probability that represents the probability of any new character

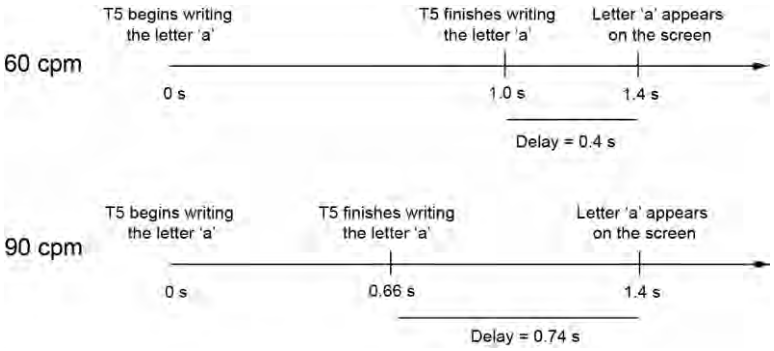
beginning at that time step. During real-time operation, we thresholded z_t (threshold = 0.3) to decide when to emit a new character. Whenever z_t crossed the threshold, we emitted the most probable character in $\boldsymbol{y}_t$ 300 ms later.

We updated the second layer of the RNN decoder at a slower frequency than the first layer (every five 20 ms time steps instead of every single time step). We found that this increased the speed and reliability of training, making it easier to hold information in memory for the length of the output delay (e.g., for a 1 second delay, the slower frequency means that the top layer must hold information in memory for only 10 steps as opposed to 50).

4.1 Estimated decoding latency

We estimate that characters were emitted to the screen between 0.4-0.7 seconds after T5 finished writing them (with the delay varying as a function of T5's writing speed). To estimate this, we first started with the fact that when T5 *begins* to write any new character at time t_0, the decoder takes 1 second to recognize this (since it is trained with a 1 second output delay). Thus, the decoder will emit a new character start signal z_t approximately 1 second after t_0. Adding an additional 0.1 seconds for the causal Gaussian smoothing delay and 0.3 seconds to emit the character after z_t crosses threshold yields a total of 1.4 seconds after t_0. This means that the new character will appear on the screen at time t_0+1.4.

To estimate the time taken between when T5 *finishes* writing a character and when it appears on the screen (which is the functional delay time of interest), we simply take 1.4 – x, where x is the time taken to write that character. For a 90 characters per minute typing rate, characters take on average 60/90 = 0.66 seconds to complete. This means the character will appear on the screen 1.4 – 0.66 = 0.74 seconds after T5 completes it (on average). For a 60 characters per minute typing rate, the delay is shorter (1.4 – 1.0 = 0.4 seconds). See the diagram below for an illustration, which shows how the delay times are calculated at these two writing speeds for the example letter 'a':



Finally, note that characters which take longer than average to write (for example, the letter 'm') will have shorter delay times.

5. RNN training overview

Here, we give an overview of the main steps and algorithms used to train the RNN (see Extended Data Fig. 2b for a diagram of the main training steps). For more details, the next section "6. RNN Training Details" provides a complete protocol, and our publicly released code implements these training methods (https://github.com/fwillett/handwritingBCI) and applies them to our publicly released data (https://doi.org/10.5061/dryad.wh70rxwmv).

Our methods are similar to neural network methods used in automatic speech recognition (Hinton et al., 2012; Graves et al., 2013; Zeyer et al., 2017; Xiong et al., 2017), but with some key changes made to achieve high performance on neural activity in a highly data-limited regime (1-10 hours of data, as opposed to 1,000+ hours (Cieri et al., 2004; Panayotov et al., 2015; He et al., 2019)).

5.1 Data labeling

A major challenge that had to be overcome for training decoders with our data is that we don't know what character T5 was writing at any given moment in time in the training data, since his hand was paralyzed. There are two major approaches used to solve this problem in automatic speech recognition: forced-alignment labeling with hidden Markov Models (HMMs) (Young et al., 2006; Hinton et al., 2012; Xiong et al., 2017), or unsupervised inference with connectionist temporal classification (Graves et al., 2006) or other similar cost functions (Collobert et al., 2016). We found that forced-alignment worked better with our data, potentially because of the relatively small dataset size. It also enabled data augmentation via synthetic sentence generation (see below). In the forced-alignment method, HMMs are used to infer what character is being written at each time step, fusing knowledge of the sequence of characters that were supposed to be written with the neural activity recorded. These character inferences can then be used to construct target probabilities that the RNN is trained to reproduce in a supervised manner.

To construct the data-labeling HMMs, we first processed the single character data to convert it into trial-averaged spatiotemporal "templates" of the neural activity patterns associated with each character. Next, these templates were used to define the emission probabilities of the HMMs, and the HMM's states and transition probabilities were set to express an orderly march through the sequence of characters in each sentence (Extended Data Fig. 2c). We then used the Viterbi algorithm to find the most probable start time of each character given the observed neural activity. The start times of each character were then used to construct target time series of character probabilities for the RNN to reproduce.

The vector of target character probabilities (denoted as $\boldsymbol{y}_t$ in section 4) was constructed by setting the probability values at each time step to be a one-hot representation of the most recently started character (i.e., the most recently started character's entry in the vector was set to 1 while all other entries were set to 0). The scalar character start probability (denoted as z_t in section 4) was set to be equal to 1 for a 200 ms window after each character began and was

otherwise equal to 0. This scalar output allows the decoder to distinguish repeated characters from single characters (e.g., “oo” vs. “o”).

One advantage of this strategy for representing the RNN output is that uncertainty about whether pauses are occurring between characters should not degrade performance, since the labeling routine only needs to identify when each character begins (not when it ends). Note that this representation causes the RNN to output a “sample-and-hold”-type signal in the $\boldsymbol{y}_t$ vector, where it will continue to output a high probability for the most recently started character until the next one begins.

5.2 Supervised training

Once the data were labeled with an HMM, we used those labels to cut out snippets of each character from the data. These snippets were then re-assembled into artificial sentences, which were added to the training data to augment it and prevent overfitting (Extended Data Fig. 2e). Although this method is simplistic in assuming that the neural representation of a character is independent of past and future characters, it was nevertheless important for achieving high performance. Including augmented data decreased the error rate percentage by 12.9 when training on single days and 2.7 when training on all days (Extended Data Fig. 3a). Finally, with the labeled and augmented dataset in hand, we used TensorFlow v1.15 (Abadi et al., 2016) to train the RNN with gradient descent using Adam (Kingma and Ba, 2017), following standard supervised training approaches.

To train the RNN to account for drifts in the *means* of neural features that naturally accrue over time in neural recordings (Downey et al., 2018; Jarosiewicz et al., 2015), we added artificial perturbations to the feature means, similar to (Sussillo et al., 2016). This step was also essential to achieving high-performance, as it decreased the error rate percentage by 5.7 (Extended Data Fig. 3b).

On each new day, we re-trained the RNN to incorporate that new day’s data before doing real-time performance evaluation. The new data were combined with all previous days’ data into one large dataset while training. To account for differences in neural activity across days (Degenhart et al., 2020; Jarosiewicz et al., 2015), we separately transformed each days’ neural activity with an affine transformation that was simultaneously optimized with the other RNN parameters (see “Combining data across days” in section 6.5 for more detail). Including multiple days of data, and fitting separate affine layers for each day, significantly improved performance (decreased the error rate percentage by 4.7 and 1.6, respectively; Extended Data Fig. 3c-d).

Hyperparameter values were largely hand-tuned; for later sessions, some parameters were tuned via small random searches over possible parameter values. Ultimately, automated parameter tuning may be required, and would certainly be useful, when applying these techniques to new participants in future clinical applications.

5.3 Unsupervised training

For the offline analysis shown in Fig. 3b, we used an un unsupervised method to train the RNN using all 50 sentences of training data collected at the beginning of each day before real-time performance evaluation. This was meant to simulate the effect of running an “in-the-background” unsupervised training routine that updates the decoder as the user writes sentences normally (and does not rely on any prior knowledge of the characters in each sentence). First, we applied the previous sessions’ RNN to the calibration data without any retraining, which simulates what would be decoded if the user begins the day with an old RNN. The output of that RNN was then fed through our language model (described in section 12) which infers the most likely character that was written at each time step given the statistics of the English language plus the RNN output. These language model inferences were then used to construct character probability targets to retrain the RNN, using the same supervised process as described above in “Supervised Training”. This procedure can be expected to improve the RNN by teaching it not to make any errors that the language model was able to properly detect and fix.

6. RNN training details

The RNN training procedure consists of four stages, as diagrammed in Extended Data Fig. 2b. Note that we have publicly released our code that implements the RNN training procedure (https://github.com/fwillett/handwritingBCI) and applies it to our publicly released data (https://doi.org/10.5061/dryad.wh70rxwmv).

The training procedure was designed to address two main challenges: (1) inferring when each character was written in the training data, so that supervised learning techniques could be used to train the RNN, and (2) augmenting the training data with synthetic sentences, to prevent the RNN from overfitting to limited data. Additionally, the training procedure differs from standard RNN training methods in that it includes day-specific input layers to account for neural activity changes across days, as well as artificial "mean drift" noise that induces robustness to within-day changes in baseline neural firing rates that occur frequently in neural recordings.

6.1 Data preprocessing

The single character data were pre-processed by binning the recorded threshold crossings into 10 ms bins. The binned rates were then z-scored (by subtracting the mean of each electrode and dividing by its bin-by-bin standard deviation) and smoothed by convolving with a Gaussian kernel (sd = 30 ms).

The sentence data were also binned (10 ms bins), smoothed by convolving with a Gaussian kernel (sd = 40 ms), and z-scored. For the sentence data, z-scoring was performed using the means and standard deviations from all trials of the *single character* data (this helps ensure that templates made using the single character data will transfer correctly to the sentences data). During real-time decoding, these same means and standard deviations were also used to z-score the data.

6.2 Stage 1: Single character time warping & averaging

For each session, at least two blocks of single character data were collected as part of the training data (10 repetitions of each character total). We then used time-warped PCA (https://github.com/ganguli-lab/twpca) (Poole et al., 2017; Williams et al., 2020) to find continuous, regularized time-warping functions to align all trials corresponding to a single character together. We used the following time-warping parameters: 5 components (temporal factors), 0.001 scale warping regularization (L1), and 1.0 scale time regularization (L2).

After time-warping, the neural activity was averaged across trials for each character, yielding an N x T matrix representing the mean neural activity pattern of each character. N is the number of microelectrodes (192) and T is the number of time steps. The time window to use for each character was chosen by visual inspection of the character shapes, using the data from session 1 (i.e., the shapes shown in Fig. 1). After the character durations were chosen, they were held fixed for all subsequent sessions. The first 100 ms of reaction time after the go cue was excluded, yielding a time window for each character that spanned 100 ms after the go cue until

the end-time chosen by inspection. Finally, the neural activity was then down-sampled to 50 ms time steps by averaging every 5 bins together, yielding single character "neural templates" that were used to build HMMs for data labeling (see below).

6.3 Stage 2: Sentence labeling with hidden Markov models

Data labeling was accomplished in 6 steps as follows:

(1) Construct an HMM for each sentence, using the neural templates from the single character data.
(2) Use the Viterbi algorithm to infer when each character began and ended in each sentence.
(3) Refine the character start times and durations using a local grid search.
(4) Update the HMM emission probabilities (but not the state transition probabilities) using the inferred character start times and durations.
(5) Repeat Steps 2-3.
(6) Construct the final targets for supervised RNN training ($\boldsymbol{y}_t$ and z_t), using the letter start times found in (5).

Step 1: HMM construction

We used hidden Markov models (HMMs) to label our data, similar to how HMMs have been used in speech recognition to determine when phonemes begin and end in an utterance where the transcription is known (this is called "forced alignment") (Young et al., 2006). For each sentence of training data, we constructed an HMM whose states and state transitions defined an orderly march through the characters of that sentence (Extended Data Fig. 2c). Each individual character was represented with a sequence of HMM states, whose multivariate Gaussian emission probabilities were determined by the single character neural templates. Specifically, for each character, the number of HMM states was equal to the number of 50 ms bins in the neural template for that character (plus an additional "blank" state), and the mean vector for each state's emission distribution was equal to the corresponding firing rate vector from the neural template. The covariance matrix was set equal to the identity matrix.

Let us denote the states of the HMM model as $s_{i,j}$, where j denotes the character number in the sentence, and i iterates through the states within each character. Table M2 below lists the state transition probabilities for states $s_{1,j}$ to $s_{N,j}$ (and the optional blank state B_j) for each character, which were hand-tuned to reasonable values. In the expressions below, N is the number of states in character j, and M is the total number of characters in the sentence.

States	Description	Transition Probabilities
$s_{x,j}$ for x<N-1	All states before the second to last, for character j	$P(s_{x,j} \rightarrow s_{x,j}) = 0.2$ $P(s_{x,j} \rightarrow s_{x+1,j}) = 0.6$ $P(s_{x,j} \rightarrow s_{x+2,j}) = 0.2$
$s_{N-1,j}$	Second to last state for character j	$P(s_{N-1,j} \rightarrow s_{N-1,j}) = 0.2$ $P(s_{N-1,j} \rightarrow s_{N,j}) = 0.8$

$s_{N,j}$	Last state for character *j*	$P(s_{N,j} \rightarrow s_{N,j}) = 0.2$ $P(s_{N,j} \rightarrow B_j) = 0.1$ $P(s_{N,j} \rightarrow s_{1,j+1}) = 0.7$
B_j	Blank state for character *j*	$P(B_j \rightarrow B_j) = 0.5$ $P(B_j \rightarrow s_{1,j+1}) = 0.5$
$s_{N,M}$	Last character state in the sentence	$P(s_{N,M} \rightarrow s_{N,M}) = 0.7$ $P(s_{N,M} \rightarrow B_M) = 0.3$
B_M	Last blank state in the sentence	$P(B_M \rightarrow B_M) = 1.0$

Table M2. Hidden Markov model parameters.

Note that B_j is a "blank" state that can be entered into at the end of the *j*th character (or skipped) and can model pauses in between characters. The emission vector for all blank states was set to the average neural activity vector across all character templates.

Step 2: Viterbi algorithm

We used the Viterbi algorithm (Rabiner, 1989) to find the most likely sequence of HMM states given the observed neural activity, with the constraint that the last state of the sequence was the last character's final state ($s_{N,M}$) or the last blank state (B_M) (this enforces that the sentence is completely finished). This constraint was implemented by setting the observation probability to zero for all other states at the final time step. We added an additional constraint that helped prevent pathological solutions: each state had to occur within a certain window of time centered on its character's location in the sentence. Let us denote the duration of the entire sentence as *T.* Each character *j*'s time window was centered on the time $(j/M)T$ and extended $0.3T$ in either direction. For example, if the character "m" occurred in the middle of the sentence, then the states for this "m" had to occur between times $0.2T$ and $0.8T$. Similarly, if the character "t" occurred in the beginning of the sentence, it had to occur between times 0 and $0.3T$. This constraint was implemented by setting a states' observation probability to zero if it lied outside of this window.

Step 3: Local refinement of character start times

The sequence of states found by the Viterbi algorithm define the start time and duration of each character. The quality of fit can be roughly assessed with correlation heatmaps that show the correlation (Pearson's *r*) between the neural template for a character and the observed neural activity, as a function of character start time and character "stretch factor" (linear time dilation factor). The identified start time and duration should lie on a hotspot (Extended Data Fig. 2d). For these heatmaps, the correlation coefficient was computed for each microelectrode channel separately; the resulting 192 correlation coefficients were then averaged together to produce a final value.

We implemented a refinement step after the Viterbi search which maximized the correlation of each character with the observed activity via a grid search of adjusted start times and template

stretch factors. The grid search varied the possible start times from 0.5 seconds before to 0.5 seconds after the HMM-identified time (in steps of 0.05 seconds). The stretch factor varied from 0.4 to 1.5 in steps of 0.0786. The stretch factor determines how the character template is contracted or dilated in time (using linear interpolation) to be longer or shorter than its average duration. Values that caused the adjusted character template to intersect adjacent character templates were not considered. This refinement procedure effectively placed each character on a nearby maximum in the heatmaps shown in Extended Data Fig. 2d.

Step 4: Updating HMM emissions

We updated the HMM emission probabilities based on the newly labeled sentence data. The emission probabilities were updated in the following way. First, for each character class, all example snippets of that character were gathered together based on the character start times and durations found above. Then, each example was time-normalized by resampling to N time steps (using linear interpolation, where N was the original number of time steps in the template). Then, the time-normalized examples were averaged together to compute a new neural template for that character. The emission probabilities were not updated for characters with less than 18 examples (e.g., rare characters such as “q” or “x”).

In principle, the state transition probabilities of the HMM could also be updated (e.g. by using the Baum-Welch algorithm). However, we did not explore that here, as we found that updating the emission probabilities alone seemed sufficient to yield high quality labels.

Step 5: Repeat with new HMM emissions

With the updated emissions, we performed one additional iteration of HMM labeling and subsequent refinement (further iterations did not seem to improve label quality).

Step 6: Construct RNN targets

Finally, target variables for supervised RNN training were generated using the letter start times found above. Two target time series were created: a series of one-hot character vectors ($\boldsymbol{y}_t$), where each vector is a one-hot representation of the most recently started character, and a scalar time series (z_t) that indicates whether *any* new character has recently been started. The z_t signal allows repeated characters to be distinguished (these would otherwise appear identical to a longer, single character as seen through $\boldsymbol{y}_t$).

Intuitively, $\boldsymbol{y}_t$ is a “sample and hold” signal that stores whatever the most recently started character was indefinitely. For example, even if T5 pauses for several seconds after writing the character “a”, $\boldsymbol{y}_t$ will still continue to reflect “a” indefinitely until a new character is started. The z_t signal is a complementary binary signal that goes high for a brief time whenever *any* new character begins. z_t can be thresholded to detect the presence of new letters and type them on the screen, which we did online. More formally, $\boldsymbol{y}_t$ and z_t were defined as follows:

$$y_{t,i} = \begin{cases} 0, & \text{the most recently started character was not i} \\ 1, & \text{the most recently started character was i} \end{cases}$$

$$z_t = \begin{cases} 0, & \text{the most recent character was started} > 200 \text{ ms ago} \\ 1, & \text{the most recent character was started} \leq 200 \text{ ms ago} \end{cases}$$

One potential advantage of this nontraditional representation is that only the character start times are required; thus, any uncertainty about when each character ends shouldn't degrade performance (i.e., uncertainty about the length of time spent transitioning between letters, either with long pauses or short bouts of pen repositioning). Additionally, by not including multiple sub-states per character (which could be an alternative way to distinguish repeated letters), this method gives the RNN freedom to decide how to break apart each character into sub-states.

6.4 Stage 3: Synthetic data generation

Once the character start times were inferred for each sentence, we generated new synthetic sentences by rearranging the characters into different sequences (Extended Data Fig. 2e). This data augmentation step improved decoding performance (Extended Data Fig. 3a), as it helped to prevent the RNN from overfitting to a limited training dataset. Here, we describe the synthesis process in detail.

Making the snippet library

First, we created a snippet library of neural activity snippets for each character. Entries were taken by extracting time windows of activity from the sentences data, beginning at the letter start time identified by the HMM procedure and ending at the start time of the next letter. In this way, any pauses and transition-related activity are included at the end of the snippets.

Generating random sentences

Next, we used the snippet library to generate synthetic sentences (24 seconds long, which was the length of data used in each minibatch during RNN training). First, the character sequence for each sentence was chosen by selecting words at random from a list of 10,000 common words (the 10,000 most frequent words appearing in the Google Web 1T 5-gram database) (Brants and Franz, 2006). Words were selected one at a time, with no dependence on the prior word, according to the following simple heuristic:

- 64% chance: a word was chosen uniformly at random from the entire list
- 20% chance: one of the twenty most frequent words was chosen uniformly at random
- 16% chance: a word with rare letters (“q”, ”x”, ”j”, or ”z”) was chosen uniformly at random from the set of all such words (this helped prevent the RNN from neglecting rare characters)

To make sure punctuation characters were represented, apostrophes were randomly added in between the last and second-to-last letter of the word (3% chance) and commas were randomly added at the end of the word (7% chance). All words either ended in a period (5% chance), question mark (5% chance), or space (90% chance).

We used the above heuristic to generate sentences instead of using real sentences in order to discourage the RNN from "baking-in" a model of the English language that extends beyond single words.

Synthesizing the neural activity

Once the synthetic character sequence was determined, the corresponding neural activity was synthesized one character at a time. For each character, a snippet was chosen from the library at random in a way that attempted to respect pen transition movements between letters. For example, when transitioning from "e" to "t", the pen must traverse upwards before beginning the downstroke for "t". However, when transitioning from "d" to "t", no such pen re-positioning is needed (when written in the way shown in Fig. 1). To do this, we discretized the starting heights for each character to the following values: 0, 0.25, 0.5, 1. The assignment of each letter to each category is depicted in Table M3 below.

Start Height	0	0.25	0.5	1.0
Character	comma	a, o, e, g, q	c, d, m, j, i, n, p, r, s, u, v, w, x, y, z, space (>), period (~)	b, t, f, h, k, l, apostrophe, question mark

Table M3. Assignment of characters to starting-height categories.

When choosing a snippet from the library, we selected at random from all snippets whose next character in the training data began at the same height as the next character in the synthetic sentence. When this wasn't possible, we selected uniformly at random from all snippets.

After a snippet was chosen, we randomly time-warped the snippet by resampling it to a different length of time (chosen uniformly from 0.7 to 1.3 times its original length). This helps the RNN to be more robust to changes in letter timing. Finally, we also sometimes added a long pause at the end of the snippet at random, to train the RNN to be robust to unpredictable pauses made by the user. The probability of adding a pause was 3%; if a pause was added, its duration was drawn from an exponential distribution (mean of 1 second). The synthetic neural activity during the pause was white noise with a standard deviation of 1.

6.5 Stage 4: Supervised RNN training

The RNN architecture itself is described above in Section 4 and illustrated in Extended Data Fig. 1. Here, we describe in detail how the RNN weights were optimized using supervised learning.

Implementation

Optimization of the RNN weights was implemented with TensorFlow v1.15 (Abadi et al., 2016). We used a desktop machine with 4 NVIDIA GeForce GTX 1080 Ti GPUs and a 32-core AMD

RYZEN Threadripper 2990WX CPU to train the RNNs. To train a single RNN, only a single GPU was used, allowing parallel training of up to 4 RNNs. A single minibatch took ~0.25 seconds to complete, resulting in training times ranging from 4 minutes (1k minibatches, when updating the RNN to a new day of data) to 3.5 hours (50k minibatches, when training from scratch). Note that the number of sentences used for training was small (572 by the last day of copy typing), so only a few minibatches were required to cycle through all sentences.

Mini-batches & gradient descent

Gradients were computed on mini-batches of 64 data snippets using backpropagation through time (Goodfellow et al., 2016). We used the gradient descent method "Adam" (Kingma and Ba, 2017) (beta1 = 0.9, beta2 = 0.999, epsilon = 0.01). The learning rate was decreased linearly from 0.01 to 0 over a pre-specified number of minibatches (1k when updating a pre-trained RNN with a new day of data, 50k when training from scratch). To prevent gradient explosion, gradient magnitudes were clipped at 10.

Each data snippet used in a mini-batch was 24 seconds long and was selected at random from the training sentences by uniformly drawing a random start time from -22 to τ-8 seconds, where τ is the duration of the sentence. Time periods outside of the sentence boundaries were not included in the cost function (by multiplying that time step's contribution by 0). Additionally, time steps corresponding to the first character at the beginning of each snippet were also not included, since the RNN might not be able to correctly identify this character if the beginning portion is not included in the snippet.

Each minibatch mixed together synthetic sentences and real sentences, in a proportion that was tuned to optimize performance (beginning at 75% synthetic and ending at ~40%). Each mini-batch selected data snippets from a single day only, which was chosen at random amongst all available days of training data. We weighted the most recent day more highly.

Cost function

We used the following cost function for a single snippet of data, which expresses the sum of an L2 weight regularization, a cross-entropy loss, and a squared error loss:

$$\lambda \sum_i \|\boldsymbol{W}_i\|_F^2 - \frac{1}{T}\sum_{t=50}^{T}\sum_{c=1}^{C} y_{t,c}\log \hat{y}_{t+d,c} + \frac{1}{T}\sum_{t=50}^{T}(z_t - \hat{z}_{t+d})^2$$

Here, λ scales the L2 regularization of the RNN weight matrices $\boldsymbol{W}_i$ (penalizing large weights), *T* is the number of time steps in the data snippet (1200, 20 ms time steps), *C* is the number of characters (31), $y_{t,c}$ is a one-hot representation of the most recently started character at time step *t*, $\hat{y}_{t+d,c}$ is the RNN's prediction of $y_{t,c}$ (*d* time steps in the future, d=50), z_t is a scalar representation of whether a character was started within the last 200 ms, and $\hat{z}_{t+d}$ is the RNN's prediction of z_t.

The one second delay (d) between the RNN output ($\hat{y}_{t+d,c}$, $\hat{z}_{t+d}$) and the target signals ($y_{t,c}$, z_t) was added to give the RNN enough time to observe all of the neural activity corresponding to a character before deciding on its identity. Note also that there is a one second "burn-in" time before the error is counted, to ensure that the RNN is not penalized for incorrectly identifying characters at the very beginning of the snippet.

Noise perturbations

We added two types of artificial noise to the neural features to regularize the RNN. First, we added white noise directly to the input feature vectors, which greatly improved performance (Extended Data Fig. 3a, middle panel). Adding white noise to the inputs asks the RNN to map clouds of similar inputs to the same output, improving generalization. The standard deviation of the white noise was an important hyperparameter that we tuned to optimize performance (see parameter values below).

We also added artificial changes to the *means* of the neural features, to make the RNN robust to non-stationarities in the neural data. Drifts in the baseline firing rates that accrue over time has been an important problem for intracortical BCIs (Jarosiewicz et al., 2015; Sussillo et al., 2016; Degenhart et al., 2020). Adding artificial mean changes greatly improved the RNN's ability to generalize to held-out blocks of data occurring later in a session (Extended Data Fig. 3b). We added two types of perturbations to the neural features to simulate non-stationarities: constant offset noise and random walk noise.

The three above-mentioned types of noise (white noise, constant offset noise and random walk noise) were all combined together to transform the input vector in the following way:

$$\acute{\boldsymbol{x}}_t = \boldsymbol{x}_t + \boldsymbol{\varepsilon}_t + \boldsymbol{\varphi} + \sum_{i=1}^{t} \boldsymbol{\upsilon}_i$$

Here, $\boldsymbol{x}_t$ are the original neural features, $\boldsymbol{\varepsilon}_t$ is a white noise vector unique to each time step, $\boldsymbol{\varphi}$ is a constant offset vector, and $\boldsymbol{\upsilon}_t$ are white noise vectors that are cumulatively summed to simulate a random walk.

Combining data across days

Combining multiple days of data together greatly improved performance relative to using just a single day (Extended Data Fig. 3c). To combine data across days, we optimized day-specific affine transforms of the input that could account for changes in the neural features across days (as observed previously in e.g. (Jarosiewicz et al., 2015; Downey et al., 2018; Degenhart et al., 2020)):

$$\tilde{\boldsymbol{x}}_t = \boldsymbol{A}_i \acute{\boldsymbol{x}}_t + \boldsymbol{b}_i$$

Here, $\acute{\boldsymbol{x}}_t$ is a vector of neural features (after artificial noise has been added, see above), $\boldsymbol{A}_i$ is a 192 x 192 matrix, and $\boldsymbol{b}_i$ is a 192 x 1 vector. The transformed features were then fed as input to the RNN. The $\boldsymbol{A}_i$ and $\boldsymbol{b}_i$ parameters are optimized simultaneously along with all other RNN parameters. Using separate input layers for each day improved performance relative to using a single shared input layer (Extended Data Fig. 3d).

Hyperparameters

We summarize the hyperparameters used for RNN training in the table below. Most parameters were hand-tuned; in later days, we performed small hyperparameter optimization routines where we trained 100 RNNs with parameters drawn at random (from pre-specified lists of reasonable values). Parameters for the next session were then taken from the top performing RNN. Thus, sometimes different hyperparameter values were used on different sessions; in particular, the amount of regularization decreased as more data were accumulated (the fraction of synthetic sentences and the white noise became smaller). Since parameters varied, we summarize their typical ranges and values in Table M4 below (rather than exhaustively list each combination for each session).

Parameter	Description	Typical Value(s)
λ	L2 Weight Regularization	1e-5
H	Hidden state size	512
σ	Standard deviation of white noise	1.0 – 1.6 (Extended Data Fig. 3a)
γ	Standard deviation of constant offset noise	0.6 (Extended Data Fig. 3b)
ζ	Standard deviation of random walk noise	0.02
χ	Fraction of synthetic trials used in each mini-batch	0.375 – 0.75 (Extended Data Fig. 3a)
ω	Probability of a min-batch drawing sentences from the most recent day	0.25 – 0.5
α	Learning rate	0.01
M	Number of min-batches to train	1k (updating for new day), 50k (training from scratch across all days)
N	Min-batch size	64
T	Duration of data snippets in each mini-batch	24 seconds

Table M4. RNN hyperparameters.

6.6 RNN parameter sweeps and variants

Retrospectively, we tested the effect of five important parameters on RNN performance (Extended Data Fig. 3): (1) the amount of synthetic data used during training, (2) the amount of

artificial white noise added to the inputs during training, (3) the amount of artificial feature *mean* noise ("drift") added during training (which simulates slow changes in baseline firing rates that accrue over time), (4) whether the RNN was trained with all days of data or just a single day, and (5) whether multiple days were combined with separate input layers or the same input layer. The results confirm that synthetic data, input white noise, feature mean noise, and combining data across days with separate input layers were all essential for high performance

When training multi-day RNNs for this analysis, we used all 10 sessions where open-loop data were collected (i.e., blocks where T5 was handwriting sentences but no real-time decoding was performed). We then evaluated performance on the 7 sessions that had both open-loop data and closed-loop copy typing data. We evaluated the character error rate on either held-out open-loop sentences ("held-out trials") or held-out closed-loop blocks ("held-out blocks") which occurred later in the session (~1 hour later). For the held-out trials, we held out 10% of the open-loop trials at random; for the held-out blocks, all data were used. When training single-day RNNs that used only a single session of data, we trained separate RNNs for all 7 evaluation sessions.

For testing the effect of synthetic data and input white noise (Extended Data Fig. 3a), we performed a joint grid search over both the synthetic data fraction and amount of white noise. Since both parameters have a regularizing effect, we searched over both at the same time so we could make more definitive statements about whether both were really needed (or whether just one tuned to the correct value provided enough regularization to reach peak performance). We searched over four possible values of the synthetic data fraction (0, 0.25, 0.5, and 0.75) and five possible values of white noise standard deviation (0, 0.6, 1.2, 1.8, and 2.4), making for a 4 x 5 grid. The character error rates shown in Extended Data Fig. 3a were averaged over the 7 evaluation days.

For testing the effect of feature mean noise (Extended Data Fig. 3b), results are shown for each of four pairs of constant offset noise (γ) and random walk noise (ζ): (γ=0, ζ=0), (γ=0.3, ζ=0.01), (γ=0.6, ζ=0.02), (γ=1.2, ζ=0.04). For each pair, 3 separate RNNs were trained.

For testing the effect of training on all days of data vs. just a single day (Extended Data Fig. 3c), results are shown from one multi-day RNN and seven single-day RNNs (one for each evaluation session).

For testing the effect of using separate input layers for each day vs. a single layer when training across multiple days (Extended Data Fig. 3d), results are shown from one separate-layer RNN and one shared-layer RNN.

6.7 Bidirectional RNN

Bidirectional RNNs are used commonly whenever the computation at hand is not required to be causal – for example, in non-real-time speech recognition or machine translation. To implement bidirectionality, we made each of the two layers bidirectional by adding an identical GRU

component that runs in the opposite direction (i.e., begins with the last neural feature vector $\tilde{\boldsymbol{x}}_T$ and ends with the first neural feature vector $\tilde{\boldsymbol{x}}_1$). The first RNN layer thus has a total of 1024 hidden units (512 units in the forward direction, 512 units in the backwards direction); these 1024 hidden units were concatenated together in a vector and fed as input to both the forward and backward components of the second layer. Likewise, the hidden state of both the forward and backward components in the second layer were concatenated together at each time step to compute the output probabilities.

Since the RNN was bidirectional, no output delay was added during training. To train the RNN, data from all available sessions were used. Instead of training and testing on separate blocks of data, as was done for the real-time performance evaluation, we used all blocks of data (both the "open-loop" blocks where no real-time decoding occurred, and the "closed-loop" blocks with real-time decoding). We randomly selected 10% of the sentences from each day as held-out test sentences for evaluation. We excluded all blocks of the 7 repeated sentences that we collected for comparison with (Pandarinath et al., 2017), since we didn't want the RNN to overfit to these sentences.

We used the following training parameters: λ=1e-5, H=512, σ=1.2, γ=0, ζ=0, χ=0.375, ω=0.1, α=0.01, M=100k, N=64, T=24.

7. Comparison to an HMM decoder

To test whether an RNN was necessary for achieving high-performance compared to a simpler decoding approach, we tested the performance of a straightforward hidden Markov model decoder (see Table M5 below). The results confirm that our RNN outperforms a simple HMM, especially for held-out blocks where the feature means are likely to have changed substantially due to neural non-stationarities (Jarosiewicz et al., 2015; Downey et al., 2018). Nevertheless, it is noteworthy that even an HMM decoder could perform reasonably well when feature mean drift was accounted for, suggesting that the neural activity itself is highly discriminable.

We designed the HMM decoder in the same way as we designed the forced-alignment HMMs used to label the sentence data, except instead of containing character states that marched forward through a fixed sequence of characters, each character could transition to any other character with equal probability. We also tweaked the state transition probabilities within each character to the following values (to improve decoding performance):

$$P(s_{x,j} \rightarrow s_{x,j}) = 0.4$$
$$P(s_{x,j} \rightarrow s_{x+1,j}) = 0.2$$
$$P(s_{x,j} \rightarrow s_{x+2,j}) = 0.4$$

When evaluating decoder performance, we applied the language model to both the RNN and the HMM. Using a language model makes the comparison fairer by compensating for the fact that the RNN itself could learn character transition probabilities that better model the English language than the simple uniform transition model we used for the HMM. In Table M5 below, offline performance (character error rate) of the RNN decoder is compared to a hidden Markov model (HMM) decoder, both with a language model applied.

	Held-Out Trials + Mean Subtraction	**Held-Out Trials**	**Held-Out Blocks**
RNN	0.23 %	0.23 %	0.70 %
HMM	2.96 %	6.70 %	80.08 %

Table M5. RNN vs. HMM Performance.

Results show that the RNN strongly outperforms the HMM, especially in situations where the feature means are likely to have changed. The HMM has no built-in mechanism for adapting to changes in the feature means (drifts in the baseline firing rates that accrue over time), leading to very poor performance when generalizing to held-out blocks, and best performance when subtracting within-block feature means to account for any changes that may have occurred over time.

8. Decoder retraining analysis (Fig. 3)

8.1 Decoder performance as a function of calibration data

To estimate how performance would have changed if we had used less calibration data to retrain the decoder each day (50 calibration sentences were originally used), we "re-ran" the copy typing sessions offline using RNNs that were trained in the same way as originally done (*except* that less calibration data were used).

First, we began with an RNN trained on our "base" dataset of 242 sentences collected on three preliminary days. Then, for each copy typing day *Y*, we retrained the RNN using *X* calibration sentences from that day (plus *X* calibration sentences from each copy typing day prior to *Y*). After retraining, we then applied the RNN to all online performance evaluation blocks. The RNN output on these blocks was used to evaluate character error rate in the same way as was done with the online data. Essentially, this procedure simulates what would have happened if the copy typing experiment were re-run in the exact same way except with a different amount of calibration data.

When reducing the amount of calibration data, we subsampled from the original 50 sentences at even intervals (thus ensuring that the subsampled data contained sentences spaced evenly in time). Note that results are similar when choosing sentences uniformly at random. To test this, we re-ran the analysis 10 more times using 10 sentences chosen randomly instead of evenly. The reported error rate in Fig. 3a was 8.5% for 10 sentences; the mean of these 10 random runs was 9.2% with a standard deviation of 0.6%.

To reduce the amount of training data even further, we also used less "single letters" data in addition to using fewer sentences. Originally, ten repetitions of each single character were collected each day as part of the decoder retraining process (these data were used to initialize the forced alignment HMM). Here, we used only 10 * (X/50) trials (i.e. 2, 4, 6, 8 and 10 trials for the 10, 20, 30, 40 and 50 sentences conditions, respectively).

Finally, for the "no retrain" condition shown in Fig. 3a-b, the inputs to the RNN were re-scaled by multiplying all input features by 1.5; this counteracts the effect of shrinkage in the original neural subspace as neural features change over time (Extended Data Fig. 4c shows the shrinkage effect).

8.2 Decoder performance as a function of days since last retrain

In Fig. 3b, we showed that less decoder retraining is needed when transferring to new days that occur closer in time (best performance was seen for 7 days or less). Eight days of copy typing data were included in this analysis, listed in the table below (see Table M1 for a more complete list of all sessions). This set of sessions were chosen so as to include as many session pairs as possible while also allowing for enough "baseline" training data for the RNN (so that it was trained on *at least* all 242 baseline training sentences before transferring to a new day).

Date (Trial Day)	Description
2019.12.09 (1209)	Real-time decoding pilot day
2019.12.11 (1211)	Copy-typing evaluation
2019.12.18 (1218)	Copy-typing evaluation
2019.12.20 (1220)	Copy-typing evaluation
2020.01.06 (1237)	Copy-typing evaluation
2020.01.08 (1239)	Copy-typing evaluation
2020.01.13 (1244)	Free-answer evaluation
2020.01.15 (1246)	Free-answer evaluation

Table M6. List of sessions included in the decoder retraining analysis (Fig. 3b).

Trial days 1209 to 1239 contained copy typing calibration blocks as well as online evaluation blocks. In this analysis, decoders were trained using some subset of the calibration data and then tested on the evaluation data. Free typing days (1244 and 1246) included copy typing data only for decoder calibration; we split this data into training and test sets so that copy typing performance could be evaluated on these days as well.

With this set of 8 sessions, we performed the following analysis for all session pairs (X, Y): train an RNN on all data from session X plus all sessions prior to X, then evaluate the RNN on session Y under different retraining conditions: no retraining, retraining with a small number of calibration sentences from session Y, or unsupervised retraining using all available calibration data from session Y.

9. Estimating neural nonstationarity (Extended Data Fig. 4)

In Extended Data Fig. 4 we showed how the recorded neural activity changed across days. Here, we explain in detail how we quantified the correlations in neural patterns across days (Extended Data Fig. 4b) and made the visualization of how neural activity contracts in the original subspace but otherwise retains similar structure (Extended Data Fig. 4c).

9.1 Neural correlations

To quantify the similarity of neural activity across days, we used correlation (Pearson's *r*) to summarize the degree of similarity across the entire neural population and all 31 characters. We used the "single letters" data collected on each day for decoder retraining.

First, threshold crossing rates were binned into 10 ms bins and smoothed by convolving with a Gaussian kernel (sd = 30 ms) to remove high frequency noise. Neural activity was then time-warped to align all repetitions of a single character together (as described in section 2.1). Neural activity was then "centered" within each day by subtracting the mean firing rate observed on each electrode. This step prevents changes in baseline firing rate from affecting the similarity measure. We wanted to exclude these, since these changes are relatively straightforward to remove/account for (as we did in this work by training the RNN to be robust to changes in mean firing rate).

Next, "pseudo-trials" were created by concatenating together a single trial from each character, resulting in pseudo-trial vectors of length NTC. Here, N is the number of electrodes (192), T is the number of time steps included from each trial (140), and C is the number of characters (31). The result of this step is a set of ten vectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_{10}\}$, one vector for each of the ten repetitions of all 31 characters. When assessing the similarity between any two days, we then have two sets of vectors to consider: $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_{10}\}$ and $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_{10}\}$. Consider each of these vectors as a random draw from a day-specific distribution (let us denote the two distributions as $\boldsymbol{V}$ and $\boldsymbol{U}$).

To quantify similarity, we estimated the correlation between the *means* of $\boldsymbol{V}$ and $\boldsymbol{U}$ (note that the means themselves are also vectors). The quantity of interest here is the *mean* because this represents the average firing rates observed for each character (i.e., the neural encoding of each character). To estimate the correlation between the means of $\boldsymbol{V}$ and $\boldsymbol{U}$, we used a cross-validated measure of correlation that reduces the impact of noise. See our prior work (Willett et al., 2020) and accompanying code repository https://github.com/fwillett/cvVectorStats for more details about this method. Importantly, this cross-validated method is different from simply correlating $\frac{1}{10}\sum_i \boldsymbol{v}_i$ and $\frac{1}{10}\sum_i \boldsymbol{u}_i$, which would underestimate the true correlation due to noise that causes the estimated means to appear more dissimilar than they really are. For example, even if $\boldsymbol{V}$ and $\boldsymbol{U}$ have identical means, noise in $\boldsymbol{v}_i$ and $\boldsymbol{u}_i$ would always cause the estimated correlation to be less than 1 when correlating $\frac{1}{10}\sum_i \boldsymbol{v}_i$ and $\frac{1}{10}\sum_i \boldsymbol{u}_i$.

To account for changes in writing speed across days, we computed the above correlation measure using varying amounts of time dilation applied uniformly to all trials from one of the days. Time dilation was implemented by linearly interpolating the neural activity to stretch or contract it in time. Ten dilation factors were considered that ranged from 0.7 to 1.42 in even increments. We used the time dilation value that maximized the correlation. This step effectively removes the potential confound of a change in writing speed causing the neural representations to appear more different than they really are.

Finally, to compute the correlation in the "anchor" space, we first projected the neural representations from each day onto the top ten principal components that describe the *earlier* day's data. This measures how much the neural activity has changed in the *original* neural subspace. The principal components were found by first concatenating all trial-averaged spatiotemporal patterns into an N x TC matrix, where N is the number of electrodes (192), T is the number of time steps (140) and C is the number of characters (31). Principal components analysis was then applied to the columns of this matrix.

9.2 Contraction in the original space

Extended Data Fig. 4c depicts a low-dimensional representation of each character's neural representation (using axes found on Day -2). To do this, each character's trial-averaged neural representation was first converted into a vector of length NT, where N is the number of electrodes (192) and T is the number of 10 ms time steps per character (140). These vectors were concatenated into a matrix of dimension NT x C, where C is the number of characters (31). Finally, principal components analysis was applied to the columns of this matrix. The top 2 PCs where then used to visualize each character's neural representation.

10. Temporal variety improves decoding (Fig. 4)

10.1 Pairwise neural distances

The characters dataset analyzed in Fig. 4 is the same as that shown in Fig. 1 (the dataset is from session 1). The straight-lines dataset was collected on a separate session (session 2 in Table M1), where T5 attempted to handwrite straight-line strokes in an instructed delay paradigm identical to what was used for writing single characters (except instead of a text cue, a line appeared on the screen to indicate the direction of the stroke).

To compute the pairwise distances reported in Fig. 4c, the threshold crossing rates were first binned into 10 ms bins and smoothed by convolving with a Gaussian kernel (sd = 30 ms). Then, neural activity within a 0 to 1000 ms window after the go cue (for characters) or 0 to 600 ms (for lines) was time-aligned across trials using the time-warping methods described above in section 2.1. These time windows were chosen by visual inspection of when the neural activity stopped modulating. The time-aligned data were then trial-averaged and re-sampled to 100 time points (using linear interpolation) to generate a set of mean spatiotemporal neural activity matrices (of dimension 192 electrodes x 100 time steps).

Pairwise distances were defined as the Euclidean norm (square root of the sum of squared entries) of the difference matrix obtained by subtracting one spatiotemporal neural matrix from another. Pairwise distances were estimated using cross-validation, according to the methods in (Willett et al., 2020) (https://github.com/fwillett/cvVectorStats); without cross-validation, noise would inflate the distances and make them all appear larger than they are. Pairwise distances for simulated data (Fig. 4f-g and Extended Data Fig. 6) were computed without cross-validation (because there was no estimation noise).

We normalized the pairwise distances reported in Fig. 4c by the number of time steps included in the analysis and the number of electrodes by dividing by $\sqrt{NT}$, where N is the number of electrodes (192) and T is the number of time steps (100). This makes the distances roughly invariant to the number of time steps and electrodes. For example, if each electrode fires at 150 Hz for condition A and 50 Hz for condition B, then the distance between B and A is $\sqrt{(150-50)^2 NT} = 100\sqrt{NT}$. Thus, dividing by $\sqrt{NT}$ would keep the distance equal to 100 for any number of time steps or electrodes.

10.2 Neural and temporal dimensionality

Dimensionality, as computed in Fig. 4e, was estimated using the "participation ratio" (Gao et al., 2017), which is a continuous metric that quantifies how evenly-sized the eigenvalues of the covariance matrix are. It is roughly equivalent to the number of dimensions needed to explain 80% of the variance in the data.

To compute the neural dimensionality, the smoothed, time-warped, and trial-averaged neural activity was arranged into a matrix **X** of dimensionality N x TC, where N is the number of electrodes (192), T is the number of time steps (100), and C is the number of movement

conditions (16). Each row is the trial-averaged response of a single electrode to each movement condition concatenated together. The eigenvalues u_i of the covariance matrix $\mathbf{X}\mathbf{X}^T$ were then used to compute the participation ratio:

$$PR = \frac{(\sum_i u_i)^2}{\sum_i u_i^2}$$

Similarly, to compute the temporal dimensionality, the neural activity was arranged into a matrix **Y** of dimensionality T x NC, where each row contains the trial-averaged response of all neurons across all conditions concatenated together for a single time step (and each column is a neural response for a single condition). Roughly, the temporal dimensionality quantifies how many T-dimensional neural response basis vectors are needed to explain 80% of the variance of all of the neural responses (PSTHs).

To reduce bias, we used cross-validation to estimate the covariance matrix (otherwise, the presence of estimation noise would artificially inflate the dimensionality). Specifically, we split the trials into two folds, and computed the **X** matrix separately for each fold (yielding $\mathbf{X}_1$ and $\mathbf{X}_2$). The covariance matrix was then estimated as $\mathbf{X}_1\mathbf{X}_2^T$. To compute confidence intervals for dimensionality (and the dimensionality ratios), we used the jackknife method (see our prior work (Willett et al., 2020) with code available at https://github.com/fwillett/cvVectorStats).

10.3 Simulated classification accuracy

To simulate classification accuracy for the lines and characters as a function of neural noise (Fig. 4d), we used the cross-validated pairwise distances between all conditions. This ensures that accuracy is not inflated by overestimating the distances between conditions. We used classic multidimensional scaling to find a set of low-dimensional points that have the same pairwise distances; these are low-dimensional representations of the neural activity patterns associated with each movement class. Then, we simulated a trial of classification with these points by first picking a point at random (the true class), and then adding Gaussian white noise to this point in the low-dimensional space (to generate the observation). Classification was correct if the observation lay closest to the true point. This simulates a simple classifier that chooses which class an observation belongs to by choosing the class with the nearest mean (this corresponds to a maximum likelihood classifier in the case of spherical Gaussian noise).

When dealing with simulated data (Fig. 4f-g or Extended Data Fig. 5-6), no multidimensional scaling was needed since no estimation noise was present. Thus, we performed the simulated classification using the true neural patterns themselves (but still in the presence of observation noise). The simulated trajectories were discretized into 100 time steps and white noise (Fig. 4, Extended Data Fig. 6) or colored noise (Extended Data Fig. 5) was added to each time step.

11. Optimized alphabet (Extended Data Fig. 6)

We considered the following optimization problem to find a new set of 26 letters that maximizes the distance between the closest pair of letters (Extended Data Fig. 6a):

$$\underset{\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_{26}}{\operatorname{argmax}} \min_{i,j} \left\| S(Q(\boldsymbol{X}_i)) - S(Q(\boldsymbol{X}_j)) \right\|_F^2$$

Here, $\boldsymbol{X}_i$ is a 2 x 100 matrix that represents the pen tip velocity trajectory for letter *i* (each column is a velocity vector for one time step), *Q* is a "squashing" function that constrains each column of $\boldsymbol{X}_i$ to lie within the unit circle, *S* is a smoothing function that constrains the trajectories to be smooth by convolving each row with a Gaussian kernel (sd = 8), and $\|\boldsymbol{X}\|_F$ is the Frobenius norm of **X** (i.e., the square root of the sum of squared entries in the matrix **X**).

Q was defined as follows (where $\boldsymbol{x}_i$ is the *i*th column of $\boldsymbol{X}$):

$$Q(\boldsymbol{X}) = [q(\boldsymbol{x}_1) \quad q(\boldsymbol{x}_2) \quad \ldots \quad q(\boldsymbol{x}_{100})]$$
$$q(\boldsymbol{x}_i) = \frac{\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|}\left(1 - e^{-\|\boldsymbol{x}_i\|}\right)$$

We found local minima of the above cost function by using gradient descent, implemented with TensorFlow v1.15 to compute the gradients. We used the "Adam" gradient descent method (Abadi et al., 2016; Kingma and Ba, 2017).

We varied the width of the Gaussian kernel used for smoothing until the temporal dimensionality of the optimized letters was equal to the dimensionality of the Latin alphabet (~4D), resulting in a set of letters that has a visually similar level of complexity and curviness to the Latin letters (Extended Data Fig. 6a). To define the pen trajectories of the Latin alphabet, we used the trajectories reconstructed in Fig. 1, which capture how T5 wrote the letters. These trajectories were resampled (linearly time-warped) to 100 time steps (as was done in Fig. 4).

As a comparison, we also optimized for a set of 26 straight lines (Extended Data Fig. 6b). To do so, we used the above cost function except that each column of $\boldsymbol{X}_i$ was constrained to be equal, enforcing a straight-line trajectory.

Note that no neural encoding model is explicitly considered in the optimization. However, if we assume linear neural tuning to pen tip velocity (i.e., "cosine tuning"), then the cost functions are equivalent under reasonable assumptions of uniformity in the neural tuning. That is, maximizing the nearest neighbor distance of pen trajectories is the same as maximizing the nearest neighbor distance of evoked neural features under this assumption. The result follows from the fact that distances are preserved under orthogonal transformations.

To show this, let $\boldsymbol{E}$ be a matrix of linear tuning coefficients (of size 192 x 2) for 192 hypothetical neural features. Assume the neural features are tuned to pen tip velocity in the following way:

$$\boldsymbol{f}_t = \boldsymbol{E}\boldsymbol{v}_t + \boldsymbol{b}$$

Here, $\boldsymbol{f}_t$ is a 192 x 1 vector of neural features for time step t, $\boldsymbol{v}_t$ is a 2 x 1 pen tip velocity vector, and $\boldsymbol{b}$ is a 192 x 1 offset vector. The squared distance between the neural feature matrices associated with two different letters can then be expressed in the following way (where $\boldsymbol{v}_t$ is the pen tip velocity for letter A and $\boldsymbol{u}_t$ is the pen tip velocity for letter B):

$$\begin{aligned}
&\sum_{t=1}^{N} (\boldsymbol{E}\boldsymbol{v}_t + \boldsymbol{b} - [\boldsymbol{E}\boldsymbol{u}_t + \boldsymbol{b}])^T (\boldsymbol{E}\boldsymbol{v}_t + \boldsymbol{b} - [\boldsymbol{E}\boldsymbol{u}_t + \boldsymbol{b}]) \\
&= \sum_{t=1}^{N} (\boldsymbol{E}(\boldsymbol{v}_t - \boldsymbol{u}_t))^T (\boldsymbol{E}(\boldsymbol{v}_t - \boldsymbol{u}_t)) \\
&= \sum_{t=1}^{N} (\boldsymbol{v}_t - \boldsymbol{u}_t)^T \boldsymbol{E}^T \boldsymbol{E} (\boldsymbol{v}_t - \boldsymbol{u}_t)
\end{aligned}$$

Let us assume that the columns of $\boldsymbol{E}$ are roughly orthogonal to each other, which would be the case for "uniform" neural tuning where the rows of $\boldsymbol{E}$ (i.e., the "preferred directions") are uniformly distributed about the origin. Then, $\boldsymbol{E}^T\boldsymbol{E}$ is a diagonal matrix with approximately equal entries along the diagonal (we can denote this entry as α). Then, we have:

$$\begin{aligned}
&\sum_{t=1}^{N} (\boldsymbol{v}_t - \boldsymbol{u}_t)^T \boldsymbol{E}^T \boldsymbol{E} (\boldsymbol{v}_t - \boldsymbol{u}_t) \\
&\approx \sum_{t=1}^{N} (\boldsymbol{v}_t - \boldsymbol{u}_t)^T \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix} (\boldsymbol{v}_t - \boldsymbol{u}_t) \\
&= \alpha \sum_{t=1}^{N} (\boldsymbol{v}_t - \boldsymbol{u}_t)^T (\boldsymbol{v}_t - \boldsymbol{u}_t)
\end{aligned}$$

Thus, linear neural tuning to pen tip velocity implies that the neural distances are directly proportional to the distances between the velocity trajectories themselves.

More generally, the neural encoding of a pen trajectory is not solely linear. However, we conjecture that as long as the neural encoding function is a reasonably smooth function of the pen tip velocity, then far-apart pen trajectories are likely to evoke far-apart neural activity, making it a reasonable approach to optimize over pen trajectories directly.

In Extended Data Fig. 6d, we show that the optimized alphabet is easier to classify than the Latin alphabet and straight-lines (for noise levels high enough to induce decoding errors).

12. Language model

12.1 Overview

In a retrospective offline analysis, we used a custom, large vocabulary language model to autocorrect errors made by the decoder. Here, we give an overview of the major steps involved (note that our code release also contains the language model and associated scripts for applying it; https://github.com/fwillett/handwritingBCI). The language model had two stages: (1) a 50,000-word bigram model that first processes the neural decoder's output to generate a set of candidate sentences, and (2) a neural network to rescore these candidate sentences (OpenAI's GPT-2, 1558M parameter version; https://github.com/openai/gpt-2) (Radford et al., 2018). This two-step strategy is typical in speech recognition (Xiong et al., 2017) and plays to the strengths of both types of models. Although the rescoring step improved performance, we found that performance was strong with the bigram model alone (1.48% character error rate with the bigram model alone, 0.89% with rescoring, using the copy typing data).

The bigram model was created with Kaldi (Povey et al., 2011) using samples of text provided by OpenAI (250k samples from WebText, https://github.com/openai/gpt-2-output-dataset). These samples were first processed to make all text lower case and to remove all punctuation that was not part of our limited character set (consisting only of periods, question marks, commas, apostrophes, and spaces). Then, we used the Kaldi toolkit to construct a bigram language model, using the 50,000 most common words appearing in the WebText sample. The language model was represented in the form of a finite-state transducer which could be used to translate the RNN probabilities into candidate sentences (Mohri et al., 2008).

12.2 WebText preprocessing

Our bigram language model was created using 250k samples of text from WebText (https://github.com/openai/gpt-2-output-dataset), provided by OpenAI (San Francisco, CA). We pre-processed the WebText samples to convert them to lowercase, remove symbols that were not in our character set, and split into sentences (yielding a total of 5.1M sentences). We applied the following step-by-step recipe to pre-process the sentences:

(1) Replace newlines and hyphens with spaces
(2) Convert all letters into lower-case
(3) Delete all characters not in the character set (which consisted only of the letters a-z, periods, spaces, commas, question marks and apostrophes)
(4) Replace repeated spaces with single spaces
(5) Remove spaces in front of periods and commas
(6) Replace repeated periods with single periods
(7) Strip surrounding whitespace from the sample
(8) Split the sample into sentences by splitting at periods and question marks

12.3 Constructing the bigram language model

We used publicly available scripts ((Puigcerver, 2017); https://github.com/jpuigcerver/Laia/tree/master/egs/iam) as a starting point for constructing our bigram language model. These scripts tokenize the list of sentences created above into discrete words, and then call Kaldi (Povey et al., 2011) and SRILM (Stolcke et al., 2011) programs to count the frequency with which these words (and pairs of words) appear in the sentences. Word frequencies are then encoded into a weighted finite state transducer (Mohri et al., 2008) that can be used to infer the most likely sequence of characters given the word frequency counts combined with the character probabilities output by the neural decoder.

To ensure that our language model was sufficiently general to allow the expression of a wide variety of sentences, we used a large vocabulary (consisting of the 50,000 most common words in the WebText samples). A space character was included at the beginning of each word to enforce that words were always separated by spaces (except for words containing punctuation, such as contractions).

In big picture, the language model is essentially a large hidden Markov model where each state corresponds to a character, and where the state transition probabilities encode the statistics of which words are likely to follow other words. Inference is done with the language model by using an approximate Viterbi search (beam search) to find likely sequences of characters. The beam search combines information from the language priors (state transition probabilities) and information from the neural decoder about which characters are likely occurring at each moment in time (observations).

The language model was represented as the composition of weighted finite state transducers (Mohri et al., 2008) that encode information about different parts of the model:

$$H \circ C \circ L \circ G$$

Here, $\circ$ denotes composition, G is the grammar that encodes legal sequences of words and their probabilities (based on the unigram and bigram probabilities), L is the lexicon that encodes what characters are contained in each legal word, and H and C encode information about the character sub-states used in decoding (see Kaldi documentation for more detail). Each character had two sub-states s_1 and s_2. s_1 emits a CTC blank (since (Puigcerver, 2017) used a neural network trained with the CTC loss function, see below) and s_2 emits the corresponding character. We used the following transition probabilities: $P(s_1 \rightarrow s_1)$=0.6, $P(s_1 \rightarrow s_2)$=0.2, $P(s_1 \rightarrow s_{next})$=0.2, $P(s_2 \rightarrow s_2)$=0.6, $P(s_2 \rightarrow s_{next})$=0.4 , where s_{next} is the first sub-state of the next character.

12.4 Inference with the bigram language model

The scripts we used from Puigcerver and colleagues (Puigcerver, 2017) configured the language model to work with outputs generated by a neural network trained with the connectionist temporal classification (CTC) loss function (Graves et al., 2006). We therefore transformed our neurally decoded probabilities to make them look more like CTC outputs before using the language model. To generate the CTC blank probability $blank_t$, we transformed z_t (the character start signal):

$$blank_t = 1 - \sigma(4 + 4\sigma^{-1}(z_{t+20}))$$

This hand-tuned function inverts z_t and makes it sharper, so that it stays mostly at 1 and dips to 0 only briefly whenever a new character is written (σ is the logistic sigmoid function). It also shifts the signal forward by 20 time steps (400 ms), so that it dips to 0 at times when the character probabilities $\boldsymbol{y}_t$ have already finished transitioning from the previous character to the next. Finally, we also modified $\boldsymbol{y}_t$ so that all entries of $\boldsymbol{y}_t$ plus the blank signal $blank_t$ sum to 1:

$$\boldsymbol{y}'_t = \boldsymbol{y}_t(1 - blank_t)$$

To perform inference with the language model and the above probabilities, we used Kaldi (and custom decoding functions by Puigcerver; https://github.com/jpuigcerver/kaldi-decoders) to perform a beam search to generate lattices of candidate word sequences with high likelihood (beam=65, max active=5000, acoustic score=1.0, lattice beam=10). On average, the decoding process completed 3.74 times faster than real-time (averaged over the last two sessions of copy typing evaluation, where T5 wrote the fastest). This suggests that it should be possible to implement the language model inference in real-time and in parallel with the RNN decoder, thereby realizing these lower error rates in real-time closed-loop sessions.

12.5 Rescoring with GPT-2

We used OpenAI's neural network language model "GPT-2" to rescore the candidate sentences inferred by the bigram language model ((Radford et al., 2018), 1558M parameter version, https://github.com/openai/gpt-2). Rescoring using a neural network model is motivated by the fact that neural networks are powerful language models that can model long-range semantic dependencies, but may be too slow to use to efficiently search through many different possibilities. Thus, a simpler N-gram model can propose a list of plausible candidate sentences which a neural network model rescores (e.g., (Xiong et al., 2017)).

When run on a sequence on characters, GPT-2 returns the conditional probability of each character given the previous characters. These probabilities can be used to compute the log probability of any candidate sentence in the following way:

$$\log P(c_1, c_2, \ldots, c_N) = \log P(c_1) + \log P(c_2 \mid c_1) + \ldots + \log P(c_N \mid c_{N-1}, \ldots, c_1)$$

Here, $P(c_1, c_2, \ldots, c_N)$ is the probability of observing the character sequence $c_1, c_2, \ldots, c_N$ and $P(c_N \mid c_{N-1}, \ldots, c_1)$ is the conditional probability of observing character c_N given previous characters $c_{N-1}, \ldots, c_1$.

When generating candidate sentences for rescoring, an "acoustic score" and a "language model score" was returned for each candidate sentence. The acoustic score contains the cumulative log probability of observing the character sequence given the sentence, and the language model score contains the log probability of observing the sentence given the language model. When rescoring, we replaced the language model score with the probability returned by GPT-2, scaled

the acoustic score by 0.5, and summed them together to generate a final score. The minimum score across all candidate sentences was then chosen.

12.6 Performance without rescoring

We compared the performance of the language model with rescoring to the bigram model alone. When decoding with the bigram model alone, we also used an acoustic score of 0.5. The character error rate with the bigram model alone was 1.48% [1.11, 1.85] (95% CI), and the word error rate was 4.72% [3.67, 5.87].

13. Statistics

The following table lists statistical details for each hypothesis test and confidence interval reported in this work. In this study, uncertainty was quantified mainly with 95% confidence intervals computed with nonparametric methods (bootstrap or jackknife). Only two hypothesis tests were performed (Fig. 4c).

Result	Statistical Details
K-NN Classifier Accuracy	The 95% confidence interval for classification accuracy reported in the main text (95% CI = [92.6, 95.8]) was a binomial proportion confidence interval (Clopper-Pearson). Classification accuracy was computed over 864 trials.
Table 1	Table 1 reports error rates derived from 163 independent trials (sentences). 95% confidence intervals were computed via bootstrap resampling over the trials (10,000 resamplings).
Fig. 3	For panel A, each data point represents the mean of 163 independent trials (sentences). 95% confidence intervals of the mean were computed via bootstrap resampling over the trials (10,000 resamples). For panel B, each data point shows the character error rate averaged across all session pairs belonging to that category. There were 8 session pairs for the 2-7 days category, 4 pairs for the 8-14 days category, and 16 pairs for the 15-37 days category. 95% confidence intervals were computed via bootstrap resampling over individual trials (10,000 resamples; resampling was performed within each session separately).
Fig. 4	For Fig. 4c, p-values were generated using two-sided, independent two-sample t-tests (with the statistical unit being a single movement condition, N=16). The assumption of normality was not quantitatively assessed. Nearest Neighbor Distance t-test: t(30)=6.86, p=1.2e-07 Mean Distance t-test: t(30)=10.83, p=6.8e-12 For Fig. 4e, the confidence intervals for spatial and temporal dimensionality were computed using the jackknife method. We used the jackknife here because the dimensionality metric was cross-validated. When dealing with cross-validated metrics, the bootstrap method can be inaccurate (see https://github.com/fwillett/cvVectorStats).

	Each jackknife subsample left out one trial from each movement condition. There were 27 trials per character condition and 24 trials per straight line condition. See (Severiano et al., 2011) for the jackknife formula, and see our prior work (Willett et al., 2020) for more examples on using the jackknife method with cross-validated metrics. Several confidence intervals were also reported in the main text (not in the figure). The confidence interval for the nearest-neighbor distance ratio (95% CI = [60%, 86%]) was computed using bootstrap resampling (10,000 resamplings of the 16 movement conditions). The confidence intervals for the dimensionality ratios (95% CI = [1.19, 1.30] and [2.58, 2.72]) were computed using the jackknife method, again because the dimensionality metric was cross-validated. Each jackknife subsample left out one trial from each movement condition.
Extended Data Fig. 3	In panel E, 95% confidence intervals for the differences in error rate between conditions were computed with bootstrap resampling of single trials (sentences). 10,000 resamplings were performed for each confidence interval. The number of trials contributing to each confidence interval varied depending on the analysis. • Synthetic data fraction – 228 trials • Input white noise – 228 trials • Feature mean noise, held-out blocks – 228 trials • Feature mean noise, held-out trials – 33 trials • Single-day vs. multi-day, held-out blocks – 228 trials • Single-day vs. multi-day, held-out trials – 33 trials • Day-specific layers vs. same layer, held-out blocks – 228 trials • Day-specific layers vs. same layer, held-out blocks – 33 trials
Extended Data Fig. 4	In panel D, each data point shows the character error rate averaged across all session pairs belonging to that category. There were 8 session pairs for the 2-7 days category, 4 pairs for the 8-14 days category, and 16 pairs for the 15-37 days category. 95% confidence intervals were computed via bootstrap resampling over individual trials (10,000 resamples; resampling was performed within each session separately).

Table M7. Statistical details for all reported confidence intervals and hypothesis tests.

II. Supplemental Note 1

In the toy example presented in Fig. 4f-h, we showed that additional temporal dimensions can be used to improve the classifiability of a set of neural patterns in the presence of Gaussian *white noise* that is uncorrelated across time points and neurons. Under these assumptions, the Euclidean distance between each pair of neural patterns is the relevant factor determining classification accuracy, and it therefore follows that greater temporal dimensionality will improve classification performance if it helps to spread out those patterns more evenly. Here, we examine how *correlated noise* might affect this result.

First, it is helpful to define some terms. Let $\boldsymbol{f}_x$ be a vector that describes the underlying neural trajectory for movement *x* (i.e., the mean neural firing rates across time for movement *x*). Each entry in the vector $\boldsymbol{f}_x$ is the mean firing rate for a single time step. To describe multiple neurons, the activity profile of each neuron is stacked one on top of the other in the vector. Let $\boldsymbol{\varepsilon}$ be a neural noise vector of the same length that has a multivariate normal distribution with zero mean and covariance matrix $\boldsymbol{\Sigma}$. If $\boldsymbol{\Sigma}$ is non-diagonal, the noise is said to be correlated.

Given a vector of noisy observed firing rates $\boldsymbol{r}= \boldsymbol{f}_x+\boldsymbol{\varepsilon}$, a maximum likelihood classifier will choose to classify $\boldsymbol{r}$ into the class that has the minimum Mahalanobis distance to $\boldsymbol{r}$ (assuming uniform class priors). In other words:

$$\underset{x}{\operatorname{argmin}}(\boldsymbol{r}-\boldsymbol{f}_x)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{r}-\boldsymbol{f}_x)$$

In the case of white noise, $\boldsymbol{\Sigma}$ is a diagonal matrix with all diagonal entries equal to σ. In this case, the classifier will simply choose the class whose mean has the smallest Euclidean distance to $\boldsymbol{r}$. This justifies the idea that nearest neighbor distances should be increased to reduce classifier confusions (potentially via spreading the neural patterns out into additional temporal dimensions).

If $\boldsymbol{\Sigma}$ is non-diagonal, this means that the noise cloud will extend more in some directions and less in others. The directions that are most harmful for classification are those that connect nearby class means (e.g., the direction $\boldsymbol{f}_y$ - $\boldsymbol{f}_x$, as this would make noise more likely to 'corrupt' class *x* to look more like *y*). In the general case where $\boldsymbol{\Sigma}$ can take any arbitrary shape, it is not always true that classification accuracy can be improved by using extra temporal dimensions to increase Euclidean distances. For example, it could be the case that these extra temporal dimensions are particularly noisy, cancelling out the benefit of increased distance between the class means. Nevertheless, under reasonable constructions of $\boldsymbol{\Sigma}$ that we test below, we show that the toy model in Fig. 4 still holds in the presence of correlated noise.

<u>Temporally Correlated Noise</u>

First, we tested noise with *temporal* correlations (meaning that the noise associated with each neuron was positively correlated in time). This noise can describe slow (but random) fluctuations in neural firing rates over time, and in this sense is more realistic than white noise. Temporal correlations would generally cause the noise to be more concentrated along dimensions that span the class means, since the underlying neural patterns are also smooth

across time (as is the case in this toy example). Extended Data Fig. 5a shows examples of temporally correlated noise vectors and the covariance matrix used to generate them. The wide diagonal band in the covariance matrix causes nearby time steps to have correlated noise.

In Extended Data Fig. 5b, we compared the classification accuracy between time-varying trajectories and constant trajectories in the presence of temporally correlated noise, finding an even more pronounced improvement for time-varying trajectories. This is because neural patterns that vary more quickly in time are less aligned with slow-varying noise directions, enabling greater robustness to this type of noise. Here, classification was performed with a maximum likelihood classifier (under the assumption that the means of each class and the covariance matrix of the noise are known). However, results also hold using a simpler "Euclidean distance" classifier that assumes the noise is white by choosing the class whose mean has the smallest Euclidean distance to $\boldsymbol{r}$.

Signal-Correlated Noise

Finally, we tested noise vectors that were directly correlated with the underlying neural signal (that is, noise vectors that contained variance *only* in signal-spanning dimensions that connect the class means, such as $\boldsymbol{f}_x$ - $\boldsymbol{f}_y$). This type of noise is more realistic than white noise in the sense that neural variability is often larger in neural dimensions that carry the signal. To find these dimensions, PCA was applied separately to the constant and time-varying trajectories to find the one (constant) or two (time-varying) spatiotemporal axes containing the neural signal. The covariance matrix was then designed to place noise in these axes only (with equal variance for each axis):

$$\boldsymbol{\Sigma} = \sigma \boldsymbol{A}\boldsymbol{A}^T$$

Here, $\boldsymbol{A}$ is a matrix whose columns are the PCA axes and σ scales the overall size of the noise. Extended Data Fig. 5c shows what these noise vectors look like for the time-varying trajectories. Because the time-varying trajectories have only two temporal dimensions, the noise vectors also have this structure (where the first 50 time points are highly correlated with each other and the last 50 time points are highly correlated with each other).

Again, even in the presence of noise that is correlated with the signal, we found that it is still easier to classify time-varying trajectories than constant trajectories (Extended Data Fig. 5d). This result can be explained by the fact that signal-spanning noise acts like white noise in dimensions that span the class means, but is zero elsewhere. Since noise in dimensions that *don't* align with the class means are not as relevant for classification performance, it makes sense that their absence does not change the main result.

III. References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. ArXiv:1603.04467 [Cs].

Brants, T., and Franz, A. (2006). Web 1T 5-gram Version 1.

Chestek, C.A., Gilja, V., Nuyujukian, P., Foster, J.D., Fan, J.M., Kaufman, M.T., Churchland, M.M., Rivera-Alvidrez, Z., Cunningham, J.P., Ryu, S.I., et al. (2011). Long-term stability of neural prosthetic control signals from silicon cortical arrays in rhesus macaque motor cortex. J. Neural Eng. *8*, 045005.

Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., and Shelhamer, E. (2014). cuDNN: Efficient Primitives for Deep Learning. ArXiv:1410.0759 [Cs].

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. ArXiv:1406.1078 [Cs, Stat].

Christie, B.P., Tat, D.M., Irwin, Z.T., Gilja, V., Nuyujukian, P., Foster, J.D., Ryu, S.I., Shenoy, K.V., Thompson, D.E., and Chestek, C.A. (2014). Comparison of spike sorting and thresholding of voltage waveforms for intracortical brain–machine interface performance. J. Neural Eng. *12*, 016009.

Cieri, C., Miller, D., and Walker, K. (2004). The Fisher Corpus: a Resource for the Next Generations of Speech-to-Text. In Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04), (Lisbon, Portugal: European Language Resources Association (ELRA)), p.

Collinger, J.L., Wodlinger, B., Downey, J.E., Wang, W., Tyler-Kabara, E.C., Weber, D.J., McMorland, A.J., Velliste, M., Boninger, M.L., and Schwartz, A.B. (2013). High-performance neuroprosthetic control by an individual with tetraplegia. The Lancet *381*, 557–564.

Collobert, R., Puhrsch, C., and Synnaeve, G. (2016). Wav2Letter: an End-to-End ConvNet-based Speech Recognition System. ArXiv:1609.03193 [Cs].

Degenhart, A.D., Bishop, W.E., Oby, E.R., Tyler-Kabara, E.C., Chase, S.M., Batista, A.P., and Yu, B.M. (2020). Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. Nature Biomedical Engineering 1–14.

Downey, J.E., Schwed, N., Chase, S.M., Schwartz, A.B., and Collinger, J.L. (2018). Intracortical recording stability in human brain–computer interface users. J. Neural Eng. *15*, 046016.

Gao, P., Trautmann, E., Yu, B., Santhanam, G., Ryu, S., Shenoy, K., and Ganguli, S. (2017). A theory of multineuronal dimensionality, dynamics and measurement. BioRxiv 214262.

Gilja, V., Nuyujukian, P., Chestek, C.A., Cunningham, J.P., Yu, B.M., Fan, J.M., Churchland, M.M., Kaufman, M.T., Kao, J.C., Ryu, S.I., et al. (2012). A high-performance neural prosthesis enabled by control algorithm design. Nat. Neurosci. *15*, 1752–1757.

Gilja, V., Pandarinath, C., Blabe, C.H., Nuyujukian, P., Simeral, J.D., Sarma, A.A., Sorice, B.L., Perge, J.A., Jarosiewicz, B., Hochberg, L.R., et al. (2015). Clinical translation of a high-performance neural prosthesis. Nat Med *21*, 1142–1145.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning (The MIT Press).

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd International Conference on Machine Learning, (Pittsburgh, Pennsylvania, USA: Association for Computing Machinery), pp. 369–376.

Graves, A., Mohamed, A., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649.

He, Y., Sainath, T.N., Prabhavalkar, R., McGraw, I., Alvarez, R., Zhao, D., Rybach, D., Kannan, A., Wu, Y., Pang, R., et al. (2019). Streaming End-to-end Speech Recognition for Mobile Devices. In ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6381–6385.

Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. IEEE Signal Processing Magazine *29*, 82–97.

Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Chen, D., Penn, R.D., and Donoghue, J.P. (2006). Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature *442*, 164–171.

Jarosiewicz, B., Sarma, A.A., Bacher, D., Masse, N.Y., Simeral, J.D., Sorice, B., Oakley, E.M., Blabe, C., Pandarinath, C., Gilja, V., et al. (2015). Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. Science Translational Medicine *7*, 313ra179-313ra179.

Kingma, D.P., and Ba, J. (2017). Adam: A Method for Stochastic Optimization. ArXiv:1412.6980 [Cs].

Maaten, L. van der, and Hinton, G. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research *9*, 2579–2605.

Masse, N.Y., Jarosiewicz, B., Simeral, J.D., Bacher, D., Stavisky, S.D., Cash, S.S., Oakley, E.M., Berhanu, E., Eskandar, E., Friehs, G., et al. (2014). Non-causal spike filtering improves decoding of movement intention for intracortical BCIs. Journal of Neuroscience Methods *236*, 58–67.

Mohri, M., Pereira, F., and Riley, M. (2008). Speech Recognition with Weighted Finite-State Transducers. In Springer Handbook of Speech Processing, J. Benesty, M.M. Sondhi, and Y.A. Huang, eds. (Berlin, Heidelberg: Springer), pp. 559–584.

Palin, K., Feit, A.M., Kim, S., Kristensson, P.O., and Oulasvirta, A. (2019). How do People Type on Mobile Devices? Observations from a Study with 37,000 Volunteers. In Proceedings of the 21st

International Conference on Human-Computer Interaction with Mobile Devices and Services, (Taipei, Taiwan: Association for Computing Machinery), pp. 1–12.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). Librispeech: An ASR corpus based on public domain audio books. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5206–5210.

Pandarinath, C., Nuyujukian, P., Blabe, C.H., Sorice, B.L., Saab, J., Willett, F.R., Hochberg, L.R., Shenoy, K.V., and Henderson, J.M. (2017). High performance communication by people with paralysis using an intracortical brain-computer interface. ELife *6*, e18554.

Poole, B., Williams, A.H., Maheswaranathan, N., Yu, B., Santhanam, G., Ryu, S.I., Baccus, S., Shenoy, K.V., and Ganguli, S. (2017). Time-warped PCA: simultaneous alignment and dimensionality reduction of neural data. Cosyne Abstracts.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., et al. (2011). The Kaldi Speech Recognition Toolkit. IEEE 2011 Workshop on Automatic Speech Recognition and Understanding.

Puigcerver, J. (2017). Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition? In 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), pp. 67–72.

Rabiner, L.R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE *77*, 257–286.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2018). Language models are unsupervised multitask learners. OpenAI Technical Report.

Severiano, A., Carriço, J.A., Robinson, D.A., Ramirez, M., and Pinto, F.R. (2011). Evaluation of Jackknife and Bootstrap for Defining Confidence Intervals for Pairwise Agreement Measures. PLOS ONE *6*, e19539.

Stolcke, A., Zheng, J., Wang, W., and Abrash, V. (2011). SRILM at Sixteen: Update and Outlook.

Sussillo, D., Stavisky, S.D., Kao, J.C., Ryu, S.I., and Shenoy, K.V. (2016). Making brain–machine interfaces robust to future neural variability. Nat Commun *7*.

Todorova, S., Sadtler, P., Batista, A., Chase, S., and Ventura, V. (2014). To sort or not to sort: the impact of spike-sorting on neural decoding performance. J. Neural Eng. *11*, 056005.

Trautmann, E.M., Stavisky, S.D., Lahiri, S., Ames, K.C., Kaufman, M.T., O'Shea, D.J., Vyas, S., Sun, X., Ryu, S.I., Ganguli, S., et al. (2019). Accurate estimation of neural population dynamics without spike sorting. Neuron.

Willett, F.R., Deo, D.R., Avansino, D.T., Rezaii, P., Hochberg, L.R., Henderson, J.M., and Shenoy, K.V. (2020). Hand Knob Area of Premotor Cortex Represents the Whole Body in a Compositional Way. Cell.

Williams, A.H., Poole, B., Maheswaranathan, N., Dhawale, A.K., Fisher, T., Wilson, C.D., Brann, D.H., Trautmann, E.M., Ryu, S., Shusterman, R., et al. (2020). Discovering Precise Temporal

Patterns in Large-Scale Neural Recordings through Robust and Interpretable Time Warping. Neuron *105*, 246-259.e8.

Xiong, W., Wu, L., Alleva, F., Droppo, J., Huang, X., and Stolcke, A. (2017). The Microsoft 2017 Conversational Speech Recognition System. ArXiv:1708.06073 [Cs].

Young, S.J., Evermann, G., Gales, M.J.F., Kershaw, D., Moore, G., Odell, J.J., Ollason, D.G., Povey, D., Valtchev, V., and Woodland, P.C. (2006). The HTK book version 3.4.

Zeyer, A., Doetsch, P., Voigtlaender, P., Schlüter, R., and Ney, H. (2017). A comprehensive study of deep bidirectional LSTM RNNS for acoustic modeling in speech recognition. In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2462–2466.

Peer Review File

Manuscript Title: High-performance brain-to-text communication via handwriting

Editorial Notes: *none*

Reviewer Comments & Author Rebuttals

Reviewer Reports on the Initial Version:

Referee #1 (Remarks to the Author):

A. Summary of the key results
This paper represents a truly novel approach to restoring communication with a brain-computer interface. Previous approaches have used point-and-click cursor control to enable communication with an onscreen keyboard and have demonstrated very good performance that enables functional performance. Here, the authors instead try to decode handwriting movements in order to predict individual letters as the brain-computer interface (BCI) users imagines writing words and sentences. Impressively, online BCI performance was more than twice as fast as previously demonstrated and approaches smartphone typing speeds. Further, the authors demonstrate that the temporal variability associated with handwriting trajectories is a major contributor to the high level of performance that was shown, which has implications for BCIs in general as it may be advantageous to try to decode complex and dexterous movements.

B. Originality and significance:
This study takes a new and original approach to BCI-controlled communication by decoding attempted handwriting movements in order to enable computer-based communication. This approach is unique because rather than decoding the movement trajectory directly (although they demonstrate that this is possible), they implement a two-step classification process using an RNN to identify when the user is attempting to write a character and then determining which character the user is trying to write. The decoding approach relies on both the spatial and temporal variability of the attempted movements to boost performance far beyond what has previously been demonstrated for BCI-based communication.

This work provides evidence that an intracortical BCI can enable fast rates of communication based on decoded handwriting patterns. This work is therefore of interest to scientists and engineers developing neural interfaces to restore communication as well as clinicians working with patients with communication impairments.

C. Data & methodology:

1) This paper is well written and clearly describes the key details and decision points that were used to implement the RNN-based decoding approach. The figures highlight key methodologicial elements and results. A rigorous approach was taken to investigate the impact of various optimization parameters, data quantity, and data quality (vs. noise). All data and code will be made publicly available providing an extremely valuable resources for the research community as well as transparency in reporting.

2) Performance metrics are appropriate and the details of how each was calculated are included.

D. Appropriate use of statistics and treatment of uncertainties:

1) All data are presented from a single subject across multiple data sessions. This is appropriate given the limited number of human participants that have been implanted with an intracortical BCI, the rigor of the approach, and the importance of the findings.

2) Statistical tests should be performed to compare between the character and lines conditions for data shown in Figure 3C and E and reflected in the manuscript and figure.

3) While many comparisons are made based on qualitative results or comparisons of confidence intervals, the effects and improvements over previous methods are large and robust. Further statistical analysis is not needed to support the conclusions.

E. Conclusions:

1) The major conclusions are robustly supported by the presented data with consistent performance achieved across multiple sessions

2) The limitations section should mention that this work comes from a single subject who had the ability to write prior to his injury.

3) The authors conclude that a handwriting BCI is the first type of BCI that has the potential to work in people with visual impairments. This was not evaluated in the present study. While the subject did not have feedback of BCI performance until after each letter was selected, this did provide feedback that could be accumulated over the course of the session. Additionally, the importance of this was not made clear. Other forms of feedback (auditory, tactile, etc.) could be used to convey information to a person with visual impairments. Further, it is a very small population that is impacted by both visual and communication impairments.

F. Suggested improvements and comments:

1) Results, line 45: specify that the participant had a cervical spinal cord injury and be more precise in the description of residual movement abilities.

2) Results, line 60: why was a non-linear approach (t-SNE) selected for data visualization and separability analysis given that PCA allowed for accurate trajectory reconstruction. Readability would be improved by understanding the intuition that guided this decision.

3) Results, line 63: Please provide a confidence interval (or similar measure of variability) for the k-nearest neighbor classification result.

4) Results, lines 95-106: It is important to note that a large amount of training data needed to be collected each day. In addition to reporting the number of sentences, the authors should report the number of characters and duration of data collection in the main text. It is noted that this information is included in the Supplemental Material. Additionally it wasn't clear from the main text that "…data was cumulatively added to the training dataset…" referred to data collected prior to BCI control, rather than just adding in data as it was collected during BCI assessment.

5) Results, figure 2C. It is interesting that day 1237 seems to have a higher character error rate that interrupts what appears to be a linear increase in error rate that is mirrored by an increase in characters per minute. Is there a reason for this? Across the 5 sessions, did the participant have a change in strategy (e.g. to go faster with less regard for error?).

6) Results, Table 1: For clarity, I suggest renaming the second row "online output + offline language model".

7) Results, Lines 166-174: How do the values chosen for simulated neural noise compare the variability in feature means that were observed in the experiment?

8) Results, Lines 178-183 & Figure 3E: While the effect of temporal dimensionality is more striking, spatial dimensionality is also likely statistically different between the characters and straight lines. This statement may therefore be too strong: "We found that the spatial dimensionality was similar for straight-lines and characters (Fig. 3E)."

9) Results- suggestions for additional data presentation:
a) Did the subject provide any subjective feedback about ease of use, training duration, suggestions for improvements, etc?
b) Had the subject previously used a point-and-click communication BCI?
c) Was there any notable change in performance within a session?
d) The authors state that the language model is capable of running in real-time. If this is the case, why wasn't this done? With the data presented, the major outcome that should be reported in the abstract is the fully-online performance with notes about how this can be improved offline.

10) Discussion, line 252: This sentence states that the subject's hand never movement, but a video is shown to highlight the micromotions. Was the subject intending to trace the letter trajectory, even if his injury likely limited his ability to do so accurately?

11) Methods, lines 689-692: Additional detail about the linear transformation and process of fitting separate input layers each day should be stated here, or clearly linked to the supplemental methods. The supplemental figure alone is not sufficient for understanding these steps.

G. References:
References are appropriate. The only comment is with regard to Reference 24 that is cited to show that EEG-BCI has achieved speeds of 60 characters per minute. This is a generous statement and other limitations could be noted given that that level of performance is not typical and was obtained from some healthy subjects due cued typing. This is a minor point.

H. Clarity and context:
1) In the abstract, results, and discussion, the authors refer to the subject as being completely paralyzed below the neck and that he performed "imagined" hand movements. However, they note that the subject retained some movement of his shoulders and that he had micromotions of his hand during the handwriting task. It would be more appropriate to describe any residual function in the subject's arm and hand. Additionally, the authors should clarify if the subject was imagining the movements or attempting them (resulting in micromotions). See for example previous work from this group: Rastogi, A., Vargas-Irwin, C.E., Willett, F.R. et al. Neural Representation of Observed, Imagined, and Attempted Grasping Force in Motor Cortex of Individuals with Chronic Tetraplegia. Sci Rep 10, 1429 (2020).

2) The abstract should report the typing speeds and accuracy that were achieved completely online without the language model since that is most representative of actual performance. It would be appropriate to also include results with offline enhancements as these would be acceptable in many contexts (such as writing an email).

Referee #2 (Remarks to the Author):

Willett et al. present an intracortical BCI (iBCI) decoding approach for classifying many characters to enable rapid typing. Their approach uses an RNN architecture to perform classification on neural activity as the subject imagines writing letters/words/sentences. They achieve typing speeds up to

90 characters per minute with above 94.5% accuracy in one subject, which significantly outperforms previous communication iBCIs. They demonstrate the system works across several sessions and both for copying text and free expression. The authors further provide analyses to provide intuition for why their approach succeeds--they achieve high classification accuracy by having the user perform a task that generates highly discriminable neural activity.

Overall, the manuscript is very well written and represents a clear and important advance in the field of BCIs. The technical innovations of the paper include 1) methods for creating training datasets when there is minimal available information (since the subject imagined moving) and 2) methods for leveraging the power of RNNs even with relatively limited data. The approaches for challenge 2 primarily use techniques common in ANNs (data augmentation) and techniques previously shown to be useful in animal studies (adding external noise to increase robustness of the networks). The solutions to challenge 1 appear relatively novel, and are certainly new to the field of BCIs. The approach/conceptual innovation of the paper is a shift away from decoding continuous control towards a method that provides accurate classification even for a relatively large 31 character set. To my knowledge this is a notable departure from prior work.

My primary concern with the manuscript is how the author's frame the work's overall approach which should more clearly emphasize the shift towards classification. As their work demonstrates, this shift can be powerful but it is also very specialized to this task. The manuscript's current comparisons to previous state-of-the-art (Pandarinath et al.) and figure 3 fail to fully make the distinction between continuous decoding of a cursor for selecting keys on a keyboard from their BCI performing a 31-way classification. Figure 3, for instance, almost implies that Pandarinath and prior BCIs were trying to classify straight line movements, which they did not. The authors' point that discriminability of the neural activity patterns directly impacts classifier performance is well taken. And provides an intuition for why having users imagine writing letters enabled their advance. But the manuscript needs to be very clear that in and of itself does not explain why they achieve higher performance. It explains why they were able to classify a large alphabet successfully for the first time. They then achieve higher performance compared to prior work because their classifier can predict letters more quickly than the average translation + click time of continuous control cursor tasks. The primary reason I emphasize this distinction is that their classification approach solves the problem of typing quite well, but does not provide a mechanism that necessarily generalizes to other tasks that are more continuous in nature like controlling a robotic limb (the authors do not claim this, but I think it's important the paper itself makes this distinction more clearly).

Specific points:
Is this the same T5 patient from Pandarinath et al. 2017? If so, it would strengthen the manuscript's claims to highlight this direct comparison (where they are also potentially at a disadvantage if studies were performed later with likely lower quality neural recordings).

If this is the same patient T5, the manuscript should mention that this subject did have the best performance of the 3-subject cohort in that prior study. While the performance advantages of their decoder are clear, given the single subject demonstration this potential subject-to-subject variability should be discussed.

The increase in characters per minute (Figure 2C) should be discussed. In addition to being more accurate over time (which may be attributed to the addition of previous day's data to the RNN training dataset), there is also an observed increase in typing speed (characters per minute). Is this also due to additional training data or other phenomena? A retrospective analysis with decoder performance on a single day's data would be useful information.

The experimental setup for real-time decoding should be clarified. Did the subject see the raw outputs during the task?

The authors nicely isolate the effect of the RNN from the more discriminable neural activity (supplemental table S4). Though I think they somewhat overstate the importance of the RNN compared to HMM in the main manuscript methods, since the RNN's main advantage is it's robustness against noise (by the authors design with noise-training for the RNN). It's actually quite noteworthy that the neural activity differences alone still lead to solid performance in a 31-way classification task with a linear HMM.

The "character stretch factor" is not well explained in the supplements. What does this factor represent?

Figure S3C and D -- are these differences statistically significant? More quantification rather than just "substantially improved" would be useful.

I'm left with an impression that many design choices in the machine learning algorithms were hand tailored. This is fine, especially for initial proof of concept. But the discussion might benefit from mentioning that methods for more automated algorithm development/training will be needed for wider utility.

Referee #3 (Remarks to the Author):

A. Summary of the key results
The work reports a single subject's performance using an intracortical BCI that can decode imagined handwriting movements from neural activity in motor cortex and map it to text in real-time. Overall the work fits within the growing body of literature intended to demonstrate faster and more accurate BCIs with improved understanding of movement encoding and more sophisticated decoding methods.
Outstanding features of the work are:
• Typing speeds of on-screen prompt at 90 characters per minute at 99% error rate with the use of a general-purpose autocorrect and 73.8 characters at 8.54% error rate for self-generated sentences (2.25% with a language model) are significant advances over the highest reported point and-click typing with a 2D computer cursor, peaking at 40 characters per minute. Results open a new approach for BCIs and demonstrate the feasibility of accurately decoding imagined handwriting movements years after losing ability to move and speak.
• The combination of probabilistic and modeling frameworks forming a hierarchical decoding approach with multiple time scales to combat neural signal variability.
• An interesting theoretical principle is proposed in which point-to-point movements may be harder to decode from one another compared to handwritten letters. Authors attribute this to the idea that temporally complex movements, such as handwriting, may be fundamentally easier to decode than point-to-point movements.

B. Originality and significance:
The paper draws upon handwriting or touch typing as a faster means to communicate by a specific population of neurologically impaired subjects. The work is an extension to this group's past contributions on BCIs for communications to the 'locked-in' population. Results presented here would be of interest to people in the BCI community who are working on restoring communication to these people who cannot move or speak.
Overall, the work is significant and original but can be better articulated. First, authors should cite the prevalence of such conditions to put this contribution in the right context.
Second, the primary performance metric is typing speed. However, on numerous occasions, the authors attempt to give the impression that this is the primary metric that could be the sole determinant for adopting the technology. While this metric is undoubtedly critical, I think the authors should reframe this argument differently, in that it is the combination of a number of

factors —one of which is typing speed— that would ultimately make the technology a first choice for the intended population. For example, the recalibration of decoders is another such factor, and while it is acknowledged by the authors that their approach is quite extensive, it is unclear how much time and resources the recalibration process takes (see detailed comments below). Another factor is the integrity of the signals over the longevity of the implant, which is a prime issue with all invasive technology (see detailed comments below).
Third, given the paper's emphasis on how the character and word decoding rates surpass existing state of the art, the data may actually have much more information about the nature of neural representation of attempted handwriting that could benefit a broader audience (particularly the neurobiology and neurophysiology communities), but this is not emphasized in the current version of the paper. As such, it is unclear if the work will be of immediate interest to many people from several disciplines.
Fourth, direct comparison to behaviors requiring dexterous movements such as typing at speeds of 120 characters per minute for intact subjects is somewhat irrelevant since the ability to modulate brain signals to become a reliable source of control of these assistive devices vary considerably among human subjects who cannot move or speak. For example, it is unclear that the achieved speed/error rates will generalize to other subjects with similar impairment. In other occasions, they draw comparison to speech-decoding BCIs for restoring verbal communication, but this technology is at a very early stage to be compared to the current approach.
Taken together, the authors should present their findings within the broader context in which the population of potential beneficiaries need to opt for a brain surgery with unknown longevity of the implanted device and a relatively long calibration process to gain additional typing speeds (extra 33 characters/min as I consider the self-paced performance reported here to be the real use case of such communication technology).

C. Data & methodology:
General comments:
The presentation is clear, logical and readable to general audience. The reporting of data and methodology is sufficiently detailed to enable reproducing the results. They state that they will share the data and code to enable reproducibility.
Major Comments:
The authors state that they 'linearly decoded pen tip velocity from neural activity'. Arguably, this variable varies considerably among different people depending on their handwriting style, accuracy, appearance, readability, etc. Did the authors have a sample handwriting from the subject before injury so they can be compared to the ones they decoded? If so, could they analyze such data to infer the pen tip speed profiles the subject likely used to better understand if the observed neural activity correlated with the character shapes? it would be more helpful if the work attempts to provide some understanding of the extent to which the dynamics of the ensemble neural activity do actually reflect this critical behavioral parameter. Also, the authors should demonstrate the extent to which character encoding might have changed as a function of trials/sentences/sessions, particularly during times when the subject was observing the prompted text, the decoded text, and when the subject was asked to write from memory. This characterization is also needed to provide credence for the claim made in the conclusion that this is a BCI without visual feedback.
It is unclear if the authors have characterized the performance long enough (beyond the stated 10 sessions) to report how nonstationarity in the neural signals can potentially deteriorate the performance reported. In fact, with the exception of the first couple of sessions that were spaced almost a month apart, the remaining 9 sessions took place almost 6 months afterwards and were closely spaced, happening within the span of 7-8 weeks. From the extensive calibration protocol described, there seems to be substantial variability in these signals.
Specific comments:
Line 93: Why did the subject write 'periods as '~' and spaces as '>'?
Line 100: Clarify if the statement 'After each new day of decoder evaluation,' refers to offline or online decoding.
Line 112: How did the authors know the exact timing of completion of each letter by the subject in

real time to be able to display it after it was completed? It is stated that visual feedback about the decoder output was 'estimated to be between 0.4-0.7'. The supplementary material explains how they arrived at these estimates, but this inherently assumes that the character was 'completed' when the start of a new one was detected. One can argue that natural handwriting of a word does not entail separating in time the representation of characters — they are all 'connected'. One can also argue that their approach (delaying the decoder output by 1 sec and adding the filter kernel widths to the total interval) prevents visual feedback about the state of neural activity until a complete character is encoded by the subject, but the reality is that the subject can 'covertly' infer information from the structure of the word being typed (self-generated case) and visual feedback from the screen (on-prompt case).
Line 115: How did neural activity look like when an error was made? and when the subject was provided visual feedback about the language model's autocorrection of that error? Did the subject stop modulating, eventually relying on the model to autocorrect, or did he continue to modulate neural activity to correct the typo? Was the decoder 'disengaged' in those instances? did the neural activity occupy different regions of the state space relative to the intended character or the corrected character?
Line 118: It is stated that the raw decoder output plateaued at 90 characters per minute with a 5.4% character error rate. But the comparison drawn in the sentence that followed argues that the 'word error rate' decreased to 3.4% average across all days. The authors should provide the reduction in 'character error rate' not 'word error rate' with the use of the language model to make this comparison objective. Arguably, many words share the same characters and understanding of words depends on the sentence context.
Line 120: it is stated that 'a new RNN was trained using all available sentences to process an entire sentence'. This means that offline decoding of an entire sentence achieved the stated 0.17% character error rate. As stated this decoder has not been used by the subject in real time to see if this newly trained decoder will be able to display an entire sentence at the end of a neural activity modulation epoch by the subject in the absence of the delayed character-by-character feedback as in the online case. As such, what is the significance of this result?
Table 1: Can the authors explain why the word error rate is so high (25.1%) in the raw online output case despite a character error rate of 5.9%?
Supplementary material:
Line 427: it is stated that "some micromotions of the right hand were visible during attempted handwriting (see 10 for neurologic exam results and SVideo 4 for hand micromotions" Have authors quantified the extent of variance in the neural data that could be explained by this potential confound?

Line 491: It would be informative for the authors to comment on how did the neural activity differ between repetitions of each character individually and when they are within a word or a sentence.

D. Appropriate use of statistics and treatment of uncertainties:
Figures are well illustrated. Probability values and error bars are explained. There were no statistical significance tests performed.

Line 178: Authors should provide more explanation for "the participation ratio (PR), which quantifies approximately how many spatial or temporal axes are required to explain 80% of the variance in the neural activity patterns" in this section. Readers have to refer to the supplementary methods section to understand this metric.

Line 192 Figure 3: The authors find that increased temporal complexity in neural state space trajectories could make movements easier to decode compared to trajectories that do not have such complexity, or have only spatial complexity. They then present a toy example in Figure 3 to make this point. I would partly disagree with their assessment and argument for the following reasons:

i) In the toy example in (Figure 3F) they increased variations of neural trajectories over time to illustrate that this increases separability (measured by nearest neighbor distance) compared to the case where the neurons' activity is constrained to a single spatial dimension, the unity diagonal). But the example lacks inclusion of noise, the temporal characteristics of which can easily 'fool' the classifier, making it think there is more temporal complexity in the trajectories than really is.
ii) The nearest neighbor distance and consequently classifier performance should be characterized when noise is present in this toy example, with a parameter that controls the amount of temporal complexity in noisy neural trajectories. Directions of fluctuations around these trajectories are likely to influence the conclusion made, both in the straight line as well as the handwritten characters cases.

Line 244: Authors state that "One unique advantage of our handwriting BCI is that, in theory, it does not require vision (since no feedback of the imagined pen trajectory is given to the participant, and letters appear only after they are completed)." I would argue against that, partially because this claim is contingent on: 1) exact knowledge of the length of time interval where each decoded character is fully known and, 2) the instructed text was always present on the screen in the on-prompt case. To my understanding this was estimated (see my comment on Line 112 above) based on approximations made by the delayed decoder training and time warping algorithm (1.4 sec delay), which was used offline to build spatiotemporal neural "templates" of the characters.

Line 534: Please clarify what is a 'single movement condition'. Is it a character, a word or a sentence? From line 801 it seems it corresponds to character but the earlier sentence needs clarification.

Line 553: Authors used character templates drawn by a computer mouse in the same way as T5 described writing the character. This description provides a shape of the character but it is unclear how this information was translated into pen velocity to train the decoder.

Line 577: "the criteria for excluding data points from display in Figure 1E is not clear. It is stated that these data labeled as "outliers in each class" were excluded "To make the t-SNE plot clearer". While it is stated that this resulted in removing 3% of data points, the explanation that these "were likely caused by lapsed attention by T5" is not convincing. How did the authors ascertain that this was the case?

Supp Fig 2 and lines 642-667: The authors use a technique from automatic speech recognition literature called forced alignment labeling with HMMs in which they augmented the data via synthetic sentence generation to cope with the limited data size. This section needs improvement regarding how the method works. For example, creating snippets to make synthetic sentences assumes the neural data correspouding to each snippet is independent of the others. How it is then integrated into a new synthetic sentence that is then labeled by the HMM? How 'one-hot representation' is defined based on the heatmaps generated in SF-2D?

E. Conclusions
The conclusions are generally based on findings in the work performed in One subject. At times though there are some overstatements about the far reaching ability of the technology which should be scaled down. For example, I did not find the conclusion that this is a BCI without visual feedback to be convincing. If it were, then how can the authors explain the difference in performance between the on-prompt typing and self-paced typing? It is unclear whether there was any type of eye tracking to determine the type of visual feedback the subject was receiving at each moment. For example, was the subject always staring at the text prompt, or was the subject always looking to the decoded characters? Or a combination of both? unless they have an objective measure of visual feedback, it is unclear whether the BCI was truly operating without vision as claimed.

F. Suggested improvements:
In addition to the above, I think a critical experiment/analysis to be performed is one in which the authors characterize the longevity and stability of representation of neural signals of the decoded variable(s). The extensive calibration process indicates that the data is highly nonstationary but none of this is characterized. Based on a few published studies, it is reasonably expected that the implanted device can leverage single cell resolution of neural spiking signals within the first year of implant. However, authors used multiunit activity (binned threshold crossing), implying the activity could not be spike sorted to reveal individual neuronal activity encoding of the pen tip velocity. More explanation should be provided on how the nonuniform distribution of session dates affected the data quality. Authors explain in the supplementary material that this approach allowed them to "leverage information from more electrodes, since many electrodes recorded activity from multiple neurons that could not be precisely spike-sorted into single units." Although they cite a paper from their group that demonstrated that neural population structure can be accurately estimated from threshold crossing rates alone, it is unclear if sorting spikes from a lower number of electrodes (which they did not state) on which single units could be identified would provide similar results.

G. References: appropriate credit to previous work?
Mostly relevant and appropriate. The work could benefit from a few more citations that documented the idea of training decoders from 'desired' behavioral templates when overt movements could not be performed.

H. Clarity and context: lucidity of abstract/summary, appropriateness of abstract, introduction and conclusions
No issues.

Author Rebuttals to Initial Comments:

Reply to Reviewers

Note: reviewers' comments appear in **black text**. Our replies appear in **blue text**, and revised manuscript text appears indented (with old text shown in **black** and new edits in **red**).

Overview:

We thank the reviewers for their careful read of the manuscript and their insightful and helpful suggestions. Most of the questions raised were requests for clarification, additional statistics, and/or reframing of certain results. We have addressed all these suggestions, which we believe has improved the presentation and rigor of the work. Point-by-point responses to each reviewer suggestion appear below this higher-level "Overview" section.

We appreciate the reviewers' unanimous recognition that this is a truly different and novel approach, with a substantial performance gain that is important for the field (and, one day hopefully, for patients as well). Three brief snippets might be helpful as it has been a while since reviewing the manuscript. Reviewer 1 (R1), "This paper represents a truly novel approach to restoring communication with a brain-computer interface." R2, "Overall, the manuscript is very well written and represents a clear and important advance in the field of BCIs." R3, "Outstanding features of the work are: Typing speeds of on-screen prompt at 90 characters per minute at 99% error rate with the use of a general-purpose autocorrect and 73.8 characters at 8.54% error rate for self-generated sentences (2.25% with a language model) are significant advances over the highest reported point and-click typing with a 2D computer cursor, peaking at 40 characters per minute. Results open a new approach for BCIs and demonstrate the feasibility of accurately decoding imagined handwriting movements years after losing ability to move and speak."

The most involved questions were raised by R3 with regards to the longevity and robustness of the intracortical BCI (iBCI). In particular, how long the neural signals can be expected to last and whether the neural signals change so quickly over time that extensive decoder retraining is required each day. With our new additions, we believe that we have addressed this question thoroughly and at a high standard, with the result being that our handwriting iBCI is right in line with other state-of-the-art iBCIs in terms of longevity and robustness. We outline below how we have addressed the longevity and robustness concerns; our additions include new discussion points as well as new data analyses that show the feasibility of achieving high-performance without requiring extensive daily decoder retraining.

Scope. Before explaining our new additions, we think it is important to first delineate what we see as the scope of this work. Any effective manuscript must have a well-defined (and necessarily limited) scope of investigation. We see this paper as being primarily focused on demonstrating the feasibility of decoding attempted handwriting movements from a person with tetraplegia well enough to substantially increase (i.e., double) communication rates while also maintaining high accuracy. Doing so opens the door to a promising new approach for iBCIs, as this is the first study to propose the fundamental idea of decoding attempted handwriting and to demonstrate that rapid sequences of attempted dexterous movements can be accurately decoded in a person who has been paralyzed for several years. However, by no means does our iBCI yet constitute a 'complete product' that would be appropriate for immediate clinical adoption, and we believe that meeting such a standard lies outside the scope of this work. Subsequent research in academia will be needed to further advance this system (e.g., just as several studies needed to follow the original Hochberg et al. *Nature* 2006 paper) and,

importantly, a truly corporate effort would be needed to fully ruggedize this, or any other, system for commercial medical use.

As suggested by R3, a final product would require systematic clinical trials that demonstrate both the longevity of the intracortical microelectrode arrays as well as decoder training algorithms that minimize (or eliminate) the need for daily decoder recalibration. To our knowledge, both "longevity" (the need to demonstrate device functionality over many years) and "robustness" (the need for less decoder retraining) are longstanding issues for intracortical BCIs, which no published manuscript has yet fully solved (but see below for reasons to be optimistic). As such, we see our work as providing important, and hopefully intellectually creative, motivation for academic researchers and companies to continue improving the longevity of intracortical arrays and designing new methods for minimizing decoder (re)calibration time. However, we do not see the complete resolution of these issues as within the scope of this study.

That said, we now outline how we have conducted extensive new data analyses and changed the manuscript to address the longevity and robustness issues to the best of our ability. We too are deeply interested in understanding these limits, so as to be most helpful to subsequent efforts.

Longevity. As R3 has noted, array longevity is a critical issue for any intracortical BCI. Before a product is taken to market, a systematic study must be conducted which demonstrates longevity across many subjects. While no such study has yet been published, preliminary results from several studies – including our own BrainGate clinical trials (NCT00912041) spanning 15 years and 14 participants – indicate that (Utah) arrays retain their functionality for several years in people; there are multiple examples of retained functionality for 1000+ days (Bullard et al., 2020; Simeral et al., 2011). Importantly, in the present study high performance was obtained 1200+ days post-implant. We added a new supplemental figure (now SFig. 6) to demonstrate that high-quality spiking activity is still present on many of the electrodes (on average 82 out of 192). In the Discussion, we now highlight the array longevity issue as well as reasons to be optimistic about array longevity.

Robustness. Second, as R3 and other reviewers have noted, minimizing decoder recalibration time is also an important problem for iBCIs (as well as non-invasive BCIs). This issue must also be addressed before a viable product can be taken to market, since users are not likely to tolerate long recalibration procedures each day. However, we see minimizing calibration time as a deep topic in and of itself, which has been the sole focus of several recent studies (Jarosiewicz et al., 2015; Dyer et al., 2017; Degenhart et al., 2020). Additionally, to our knowledge, daily decoder recalibration is still standard practice in the iBCI field and many important papers have used this method (e.g., Hochberg et al., 2006, 2012; Collinger et al., 2013; Bouton et al., 2016; Ajiboye et al., 2017). We think it is therefore reasonable to leave this aspect of handwriting decoding to be more fully investigated in future work. *Nevertheless*, we agree that it is important to both (1) more clearly highlight this issue in the manuscript, and (2) do whatever analyses we can to address it while still remaining reasonably within scope.

To that end, **we have added a new figure to the main text (now Fig. 3)** that reports results from offline analyses estimating how much data was actually needed for daily decoder retraining. Encouragingly, the results suggest that high performance would have been possible with only 10 sentences of data per day (as opposed to the 50 sentences per day that were originally used). We also report promising results from a new unsupervised method, that we

introduce here in the revised manuscript thanks to the Reviewers' questions, that uses a language model to retrain the decoder without requiring any explicit data labels. This could enable decoder retraining to occur in the background, as a parallel computational process making use of newly incoming data, without interrupting the person's iBCI use. We believe these analyses show promise that it should be possible to achieve high performance with unsupervised retraining, combined with smaller amounts of supervised data after long periods of not using the iBCI. This points the way towards a handwriting iBCI that can achieve high performance while minimizing user interruptions.

We also added a new supplemental figure (now SFig. 4) that assess the stability of the neural patterns associated with each character over time, since this is a critical issue that ultimately determines how much data is needed for daily decoder recalibration. We found high short-term stability (mean correlation = 0.85 when 7 days apart or less), and neural changes that seemed to accumulate at a steady and predictable rate. Again, this is promising for the possibility of recalibrating decoders with limited amounts of data (or even in an unsupervised manner without interrupting the user).

Future Work. Although we cannot fully resolve the longevity and robustness issues in this current manuscript, we do want the Reviewers and Editors to know that we appreciate the importance of these issues in general. As such, we thought it might be helpful to share that we are currently in the process of writing a separate manuscript summarizing array safety and longevity data from all 14 participants of the BrainGate pilot clinical trial (collected over a span of 15 years), which will be the first systematic study of its kind in people. We think that this kind of a study is a better forum for more fully resolving these issues than what this current manuscript can do, which we think should remain focused on laying out and demonstrating an entirely new kind of iBCI and associated methods.

References

Ajiboye, A.B., Willett, F.R., Young, D.R., Memberg, W.D., Murphy, B.A., Miller, J.P., Walter, B.L., Sweet, J.A., Hoyen, H.A., Keith, M.W., Peckham, P.H., Simeral, J.D., Donoghue, J.P., Hochberg, L.R., Kirsch, R.F., 2017. Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration. The Lancet 389, 1821–1830. https://doi.org/10.1016/S0140-6736(17)30601-3

Bouton, C.E., Shaikhouni, A., Annetta, N.V., Bockbrader, M.A., Friedenberg, D.A., Nielson, D.M., Sharma, G., Sederberg, P.B., Glenn, B.C., Mysiw, W.J., Morgan, A.G., Deogaonkar, M., Rezai, A.R., 2016. Restoring cortical control of functional movement in a human with quadriplegia. Nature 533, 247–250. https://doi.org/10.1038/nature17435

Bullard, A.J., Hutchison, B.C., Lee, J., Chestek, C.A., Patil, P.G., 2020. Estimating Risk for Future Intracranial, Fully Implanted, Modular Neuroprosthetic Systems: A Systematic Review of Hardware Complications in Clinical Deep Brain Stimulation and Experimental Human Intracortical Arrays. Neuromodulation Technol. Neural Interface 23, 411–426. https://doi.org/10.1111/ner.13069

Collinger, J.L., Wodlinger, B., Downey, J.E., Wang, W., Tyler-Kabara, E.C., Weber, D.J., McMorland, A.J., Velliste, M., Boninger, M.L., Schwartz, A.B., 2013. High-performance neuroprosthetic control by an individual with tetraplegia. The Lancet 381, 557–564. https://doi.org/10.1016/S0140-6736(12)61816-9

Degenhart, A.D., Bishop, W.E., Oby, E.R., Tyler-Kabara, E.C., Chase, S.M., Batista, A.P., Yu, B.M., 2020. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. Nat. Biomed. Eng. 1–14. https://doi.org/10.1038/s41551-020-0542-9

Dyer, E.L., Gheshlaghi Azar, M., Perich, M.G., Fernandes, H.L., Naufel, S., Miller, L.E., Körding, K.P., 2017. A cryptography-based approach for movement decoding. Nat. Biomed. Eng. 1, 967–976. https://doi.org/10.1038/s41551-017-0169-7

Hochberg, L.R., Bacher, D., Jarosiewicz, B., Masse, N.Y., Simeral, J.D., Vogel, J., Haddadin, S., Liu, J., Cash, S.S., Smagt, P. van der, Donoghue, J.P., 2012. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. Nature 485, 372–375. https://doi.org/10.1038/nature11076

Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Chen, D., Penn, R.D., Donoghue, J.P., 2006. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature 442, 164–171. https://doi.org/10.1038/nature04970

Jarosiewicz, B., Sarma, A.A., Bacher, D., Masse, N.Y., Simeral, J.D., Sorice, B., Oakley, E.M., Blabe, C., Pandarinath, C., Gilja, V., Cash, S.S., Eskandar, E.N., Friehs, G., Henderson, J.M., Shenoy, K.V., Donoghue, J.P., Hochberg, L.R., 2015. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. Sci. Transl. Med. 7, 313ra179-313ra179. https://doi.org/10.1126/scitranslmed.aac7328

Simeral, J.D., Kim, S.-P., Black, M.J., Donoghue, J.P., Hochberg, L.R., 2011. 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. https://doi.org/10.1088/1741-2560/8/2/025027

Point-by-point responses to referee #1

A. Summary of the key results
This paper represents a truly novel approach to restoring communication with a brain-computer interface. Previous approaches have used point-and-click cursor control to enable communication with an onscreen keyboard and have demonstrated very good performance that enables functional performance. Here, the authors instead try to decode handwriting movements in order to predict individual letters as the brain-computer interface (BCI) users imagines writing words and sentences. Impressively, online BCI performance was more than twice as fast as previously demonstrated and approaches smartphone typing speeds. Further, the authors demonstrate that the temporal variability associated with handwriting trajectories is a major contributor to the high level of performance that was shown, which has implications for BCIs in general as it may be advantageous to try to decode complex and dexterous movements.

B. Originality and significance:
This study takes a new and original approach to BCI-controlled communication by decoding attempted handwriting movements in order to enable computer-based communication. This approach is unique because rather than decoding the movement trajectory directly (although they demonstrate that this is possible), they implement a two-step classification process using an RNN to identify when the user is attempting to write a character and then determining which character the user is trying to write. The decoding approach relies on both the spatial and temporal variability of the attempted movements to boost performance far beyond what has previously been demonstrated for BCI-based communication.

This work provides evidence that an intracortical BCI can enable fast rates of communication based on decoded handwriting patterns. This work is therefore of interest to scientists and engineers developing neural interfaces to restore communication as well as clinicians working with patients with communication impairments.

We are gratified that the reviewer expresses that this is a truly novel approach that makes significant gains in intracortical brain-computer interface (iBCI) performance. We thank the reviewer for their thorough read of the manuscript and insightful and helpful questions and suggestions.

C. Data & methodology:

1) This paper is well written and clearly describes the key details and decision points that were used to implement the RNN-based decoding approach. The figures highlight key methodologicial elements and results. A rigorous approach was taken to investigate the impact of various optimization parameters, data quantity, and data quality (vs. noise). All data and code will be made publicly available providing an extremely valuable resources for the research community as well as transparency in reporting.

Thank you for noting the methodological rigor and the value of the data & code release, which we too think will help the research community improve upon what we have done and apply our methods to new problems.

2) Performance metrics are appropriate and the details of how each was calculated are included.

D. Appropriate use of statistics and treatment of uncertainties:

1) All data are presented from a single subject across multiple data sessions. This is appropriate given the limited number of human participants that have been implanted with an intracortical BCI, the rigor of the approach, and the importance of the findings.

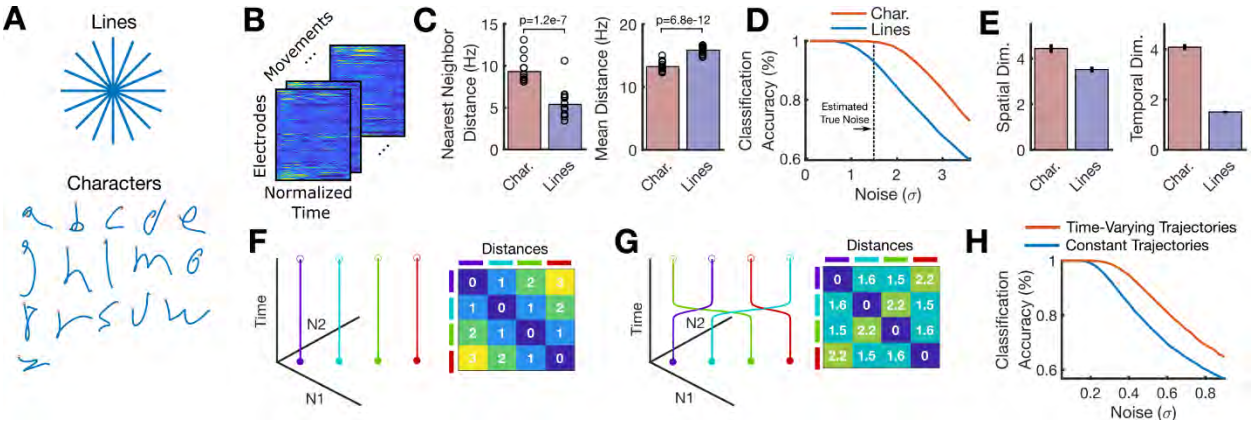
Thank you for explicitly noting that one subject is appropriate for this type of study. We too believe this to be the case.

2) Statistical tests should be performed to compare between the character and lines conditions for data shown in Figure 3C and E and reflected in the manuscript and figure.

Thank you for this suggestion. We now report 95% confidence intervals for the effects shown in Fig. 3C and 3E (which is now Fig. 4). We believe that confidence intervals keep the focus on the effect sizes, while also demonstrating statistical significance. Confidence intervals were generated by jackknife. Below is a snippet from the main text where the confidence intervals were added:

> First, we analyzed the pairwise Euclidean distances between each neural activity pattern. We found that the nearest-neighbor distances for each movement were ~~almost twice as large~~ 72% larger for characters as compared to straight lines (95% CI = [60%, 86%])~~(72% larger)~~, making it less likely for a decoder to confuse two nearby characters (Fig. 4~~3~~C).
>
> …
>
> We found that ~~the~~ spatial dimensionality was ~~similar for straight lines and characters~~only modestly larger for characters (~~Fig. 3E~~1.24 times larger; 95% CI = [1.11, 1.37]), but that the temporal dimensionality was much greater (~~more than twice~~ 2.65 times larger; ~~as large for characters~~ 95% CI = [2.63, 2.68]), suggesting that the increased variety of temporal patterns in letter writing drives the increased separability of each movement (Fig. 4E).

Using two-sample t-tests, we also now report p-values for Fig. 4C (see below). The t-tests compare the means of the distributions shown (n=16).



Finally, it does not seem straightforward to apply hypothesis testing to 4E, since temporal and spatial dimensionalities are complex functions of the data that do not appear to have standard tests or null distributions. We believe that the 95% confidence intervals shown on the bars (computed via jackknife), as well as the new confidence intervals for the dimensionality ratios mentioned above, sufficiently demonstrate statistical significance.

3) While many comparisons are made based on qualitative results or comparisons of confidence intervals, the effects and improvements over previous methods are large and robust. Further statistical analysis is not needed to support the conclusions.

Thank you.

E. Conclusions:

1) The major conclusions are robustly supported by the presented data with consistent performance achieved across multiple sessions

Thank you.

2) The limitations section should mention that this work comes from a single subject who had the ability to

write prior to his injury.

Thank you, we have now added this limitation to the Discussion section:

> Finally, it is important to recognize that ~~our~~ the current system is a proof-of-concept that a high-performance handwriting BCI is possible (in a single participant capable of handwriting prior to his injury); it is not yet a complete, clinically viable system.

3) The authors conclude that a handwriting BCI is the first type of BCI that has the potential to work in people with visual impairments. This was not evaluated in the present study. While the subject did not have feedback of BCI performance until after each letter was selected, this did provide feedback that could be accumulated over the course of the session. Additionally, the importance of this was not made clear. Other forms of feedback (auditory, tactile, etc.) could be used to convey information to a person with visual impairments. Further, it is a very small population that is impacted by both visual and communication impairments.

Thank you for raising these important limitations. We agree, and now no longer discuss our iBCI's potential to work in people with visual impairments. While we did collect some data demonstrating good performance with his eyes closed that could be added, it is not a major point and we believe that it is better to remove it to help the manuscript stay focused.

F. Suggested improvements and comments:

1) Results, line 45: specify that the participant had a cervical spinal cord injury and be more precise in the description of residual movement abilities.

The description now reads:

> T5 has a high-level spinal cord injury (C4 AIS C) and was paralyzed from the neck down; his hand movements were entirely non-functional and limited to twitching and micromotion.

Also, note that in the Methods section we refer to T5's neurologic exam data that was recently published as part of a different paper (Willett et al. *Cell* 2020, cited below). We have added more detail to the Methods section which now reads as follows:

> Below the injury, T5 retained some very limited voluntary motion of the arms and legs that was largely restricted to the left elbow; however, some micromotions of the right hand were visible during attempted handwriting (see [12] for full neurologic exam results and SVideo 4 for hand micromotions). T5's neurologic exam findings were as follows for muscle groups controlling the motion of his right hand: Wrist Flexion=0, Wrist Extension=2, Finger Flexion=0, Finger Extension=2 (MRC Scale: 0=Nothing, 1=Muscle Twitch but no Joint Movement, 2=Some Joint Movement, 3=Overcomes Gravity, 4=Overcomes Some Resistance, 5=Overcomes Full Resistance).

[12] Willett FR, Deo DR, Avansino DT, Rezaii P, Hochberg LR, Henderson JM, Shenoy KV (2020) Hand Knob Area of Premotor Cortex Represents the Whole Body in a Compositional Way. Cell 181:396–409.

2) Results, line 60: why was a non-linear approach (t-SNE) selected for data visualization and separability analysis given that PCA allowed for accurate trajectory reconstruction. Readability would be improved by understanding the intuition that guided this decision.

Thank you for raising this lack of clarity. The difference is that the trajectory reconstruction was performed on the trial-averaged data (which averages and thereby reduces noise), while t-SNE was applied to single trials (which inevitably have considerable noise). The advantage of t-SNE (as compared to a method like PCA) is that t-SNE is designed to accurately portray the separability of high-dimensional clusters in the presence of noise, while PCA on single trials will often show highly overlapping clusters in low-

dimensional space that are highly separable in the full-dimensional space. To make this clearer, we now emphasize more clearly that the trajectory reconstruction was performed on trial-averaged data, while t-SNE was applied to single trials (originally this was spelled out only in the figure legend):

> To see if the neural activity encoded the pen movements needed to draw each character's shape, we attempted to reconstruct each character by linearly decoding pen tip velocity from the trial-averaged neural activity (Fig. 1D). Readily recognizable letter shapes confirm that pen tip velocity is robustly encoded.
> ….
> Finally, we used a nonlinear dimensionality reduction method (t-SNE) to produce a 2-dimensional visualization of each single trial's neural activity recorded during a 1 s window after the 'go' cue was given (Fig. 1E).

3) Results, line 63: Please provide a confidence interval (or similar measure of variability) for the k-nearest neighbor classification result.

A confidence interval is now provided (binomial proportion confidence interval, Clopper-Pearson method):

> Using a k-nearest neighbor classifier applied to the neural activity, we could classify the characters with 94.1% accuracy (95% CI = [92.6, 95.8], chance level = 3.2%).

4) Results, lines 95-106: It is important to note that a large amount of training data needed to be collected each day. In addition to reporting the number of sentences, the authors should report the number of characters and duration of data collection in the main text. It is noted that this information is included in the Supplemental Material. Additionally it wasn't clear from the main text that "…data was cumulatively added to the training dataset…" referred to data collected prior to BCI control, rather than just adding in data as it was collected during BCI assessment.

Thank you for raising this important point. We have added the above-requested details and rephrased the main text to clarify how the training data were used. The description now reads:

> Prior to the first day of real-time use described here, we collected a total of 242 sentences across 3 days that were combined to train the RNN (sentences were selected from the British National Corpus). On each day of real-time use, additional training data were collected to retrain the RNN prior to real-time evaluation, yielding a combined total of 572 training sentences by the last day (comprising 7.3 hours and 30.4k characters). ~~After each new day of decoder evaluation, that day's data was cumulatively added to the training dataset for the next day (yielding a total of 572 sentences by the last day).~~

In addition, based on feedback from the other reviewers, we have added new offline analyses that estimate how much data were actually needed for daily decoder retraining. The results suggest that high performance is possible with only 10 sentences of data per day (as opposed to the 50 sentences per day that were originally used). We also report results from a new unsupervised method that uses a language model to retrain the decoder without requiring any explicit data labels; this could enable decoder retraining to occur in the background without interrupting the user for data collection. We believe these analyses both (1) draw important attention to this issue, and (2) show promise that it may be possible to achieve high performance with unsupervised retraining (combined with smaller amounts of supervised data after long periods of not using the iBCI). This points the way towards a handwriting iBCI that can achieve high performance while minimizing user interruptions.

For convenience, this new Results section is reproduced below:

> Following standard practice for BCIs (e.g. [1,2,19,4,5]), we retrained our handwriting decoder each day before evaluating it, with the help of "calibration" data collected at the beginning of each day. Retraining helps account for changes in neural recordings that accrue over time. Ideally, to reduce the burden on the user, little or no calibration data would be required. In a retrospective analysis of the copy typing data reported

above in Fig. 2, we assessed whether high performance could still have been achieved using less than the original 50 calibration sentences per day (Fig. 3A). We found that 10 sentences (8.7 minutes) were enough to achieve a raw error rate of 8.5% (1.7% with a language model), although 30 sentences (26.1 minutes) were needed to match the raw online error rate of 5.9%.

However, our copy typing data were collected over a 28-day time span, possibly allowing larger changes in neural activity to accumulate. We therefore tested whether more closely-spaced sessions reduce the need for calibration data (Fig. 3B), using an offline analysis of copy typing data across 8 sessions. We found that when only 2-7 days passed between sessions, performance was reasonable with *no* decoder retraining (11.1% raw error rate, 1.5% with a language model). Finally, we tested whether decoders could be retrained in an unsupervised manner by using a language model to error-correct and retrain the decoder, thus bypassing the need to interrupt the user for calibration (i.e. by recalibrating automatically during normal use). Encouragingly, unsupervised retraining achieved a 7.3% raw error rate (0.84% with a language model) when sessions were separated by 7 days or less (see Methods & Supplemental Methods for details). Ultimately, whether decoders can be successfully retrained with minimal recalibration data depends on how quickly the neural activity changes over time. We assessed the stability of the neural patterns associated with each character and found high short-term stability (mean correlation = 0.85 when 7 days apart or less), and neural changes that seemed to accumulate at a steady and predictable rate (SFig. 4 provides a quantitative visualization). The above results are promising for clinical viability, as they suggest that unsupervised decoder retraining, combined with more limited supervised retraining after longer periods of inactivity, may be sufficient to achieve high performance.

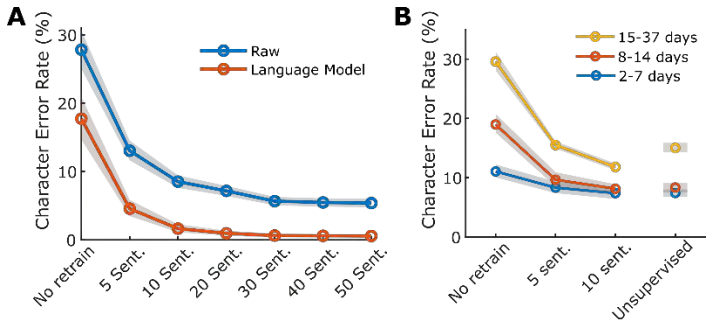


Figure 3. Performance remains high when decoder retraining is limited or omitted. (A) To account for neural activity changes that accrue over time, we retrained our handwriting decoder each day before evaluating it. Here, we simulate offline what the decoding performance shown in Fig. 2 would have been if less than 50 calibration sentences were used. Lines show the mean error rate across all data and shaded regions indicate 95% CIs (computed via bootstrap resampling of single trials, N=10,000). **(B)** Copy typing data from eight sessions were used to assess whether less calibration data are required if sessions occur closer in time. All session pairs (X, Y) were considered. Decoders were first initialized using training data from session X and earlier, and then evaluated on session Y under different retraining methods (no retraining, retraining with limited calibration data, or unsupervised retraining). The average raw character error rate is plotted for each category of time elapsed between sessions X and Y, and for each retraining method. Shaded regions indicate 95% CIs.

5) Results, figure 2C. It is interesting that day 1237 seems to have a higher character error rate that interrupts what appears to be a linear increase in error rate that is mirrored by an increase in characters per minute. Is there a reason for this? Across the 5 sessions, did the participant have a change in strategy (e.g. to go faster with less regard for error?).

Day 1237 does indeed seem to be an outlier, but we don't have a strong reason to suspect a particular cause. T5 was always instructed to go as fast as possible; anecdotally, he reported becoming more comfortable with going faster over time, as he gained confidence that the system would work accurately at higher speeds. There is indeed a modest increase in error rate over time (we observed an error rate of 4.3% on the first day and 5.4% on the last day). We speculate that it is easier to classify at slower speeds

because of an increased amount of neural data per character that can be used to decide that character's identity, effectively increasing the overall SNR of the data available to the decoder.

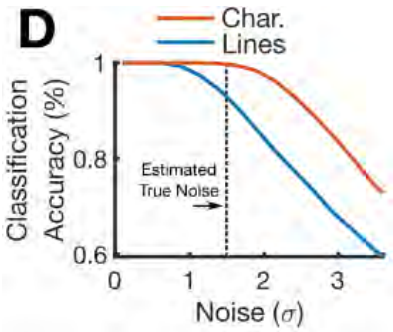
6) Results, Table 1: For clarity, I suggest renaming the second row "online output + offline language model".

Thank you for this suggestion. We have reformatted the table as follows:

	Character Error Rate (%) [95% CI]	Word Error Rate (%) [95% CI]
Online output	5.9 [5.3, 6.5]	25.1 [22.5, 27.4]
Online output + offline language model	0.89 [0.61, 1.2]	3.4 [2.5, 4.4]
Offline bidirectional RNN + language model	0.17 [0, 0.36]	1.5 [0, 3.2]

7) Results, Lines 166-174: How do the values chosen for simulated neural noise compare the variability in feature means that were observed in the experiment?

We have now annotated the figure with an estimate of the true noise level in the recorded neural features:



This estimate was generated by computing the neural population distance of each single trial from the class mean, along neural dimensions that connect each class to each other class (thus ignoring dimensions that are irrelevant for classification).

8) Results, Lines 178-183 & Figure 3E: While the effect of temporal dimensionality is more striking, spatial dimensionality is also likely statistically different between the characters and straight lines. This statement may therefore be too strong: "We found that the spatial dimensionality was similar for straight-lines and characters (Fig. 3E)."

Thank you for pointing this out. We now report the results as follows, which do indeed reveal a modest (but statistically significant) increase in spatial dimensionality for characters:

> We found that ~~the~~ spatial dimensionality was ~~similar for straight-lines and characters~~only modestly larger for characters (~~Fig. 3E~~1.24 times larger; 95% CI = [1.11, 1.37]), but that the temporal dimensionality was much greater (~~more than twice~~ 2.65 times larger; ~~as large for characters~~ 95% CI = [2.63, 2.68]), suggesting that the increased variety of temporal patterns in letter writing drives the increased separability of each movement (Fig. 4E).

9) Results- suggestions for additional data presentation:

a) Did the subject provide any subjective feedback about ease of use, training duration, suggestions for improvements, etc?

One of the most interesting things T5 described to us was his own experience of what it felt like to 'attempt' to handwrite. T5 imagined that he was holding a pen in his hand. As he attempted to write each letter, he reported having the subjective experience of feeling as though the pen was actually moving and tracing out the letter shapes (even though the actual motion of his hand was very limited, and he was not holding a pen). He sometimes reported being reticent about writing more quickly, because this could cause the subjective experience to lose clarity. He also reported that this experience seemed to follow physical constraints, because he was able to 'write' more quickly if he attempted to write in a smaller font. We now mention this subjective experience in the Results section, which reads:

> We instructed T5 to 'attempt' to write as if his hand was not paralyzed (while imagining that he was holding a pen on a piece of ruled paper). T5 reported having the subjective experience of feeling as though the imaginary pen was moving and tracing out the letter shapes.

Regarding suggestions for improving the BCI, T5 did not have much to say. Mostly, T5 was happy and somewhat amazed that the BCI could figure out what he was writing and show it to him on the screen. T5 reported feeling like he wasn't making 'clear' or 'legible' movements, and so he was surprised at how consistently the BCI could decode what he was trying to write.

b) Had the subject previously used a point-and-click communication BCI?

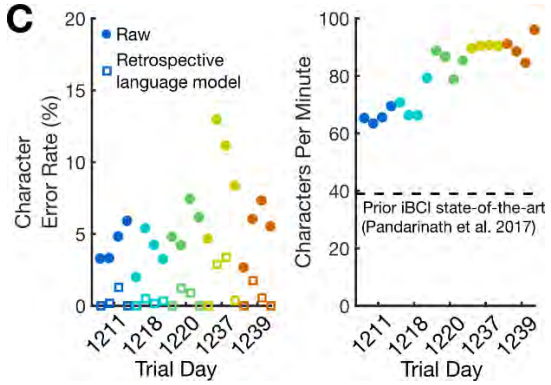
Yes, T5 set the prior record for BCI communication using a point-and-click BCI in one of our prior publications (Pandarinath et al. *eLife* 2017, cited below). We now explicitly mention this in the Results section:

> For intracortical BCIs, the highest performing method has been point-and-click typing with a 2D computer cursor, peaking at 40 characters per minute [7] (this record was also set by participant T5 3 years earlier; see SVideo 3 for a direct comparison).

Pandarinath C, Nuyujukian P, Blabe CH, Sorice BL, Saab J, Willett FR, Hochberg LR, Shenoy KV, Henderson JM (2017) High performance communication by people with paralysis using an intracortical brain-computer interface. Elife 6 Available at: http://dx.doi.org/10.7554/eLife.18554.

c) Was there any notable change in performance within a session?

Thank you for this interesting suggestion. Fig. 2C (reproduced below) shows a relatively flat character error rate within each session (as can be assessed by looking at the four circles from each session, each of which corresponds to a single block of sentences). Nevertheless, there does appear to be a modest (but potentially statistically significant) increase in error near the end of each session.



Since the first and last block contain the same seven 'comparison' sentences which we collected for a direct comparison to prior work (Pandarinath et al. 2017), we can directly compare the difference in error between the first and last block of each session. Pooling all the data together reveals an increase in error

rate of 2.4% (95% CI = [0.63, 4.2]). However, it is difficult to know whether this increase in error is due to small changes in neural activity which accrue over time, or due to the participant's fatigue after a long session. We think that an interesting area of future work could attempt to iteratively adjust the decoder to account for neural changes after each sentence is decoded (using unsupervised retraining), to see if this prevents the error rate from increasing. However, as there is already a lot to discuss in the paper, we think it is best to tackle this issue in a separate study.

d) The authors state that the language model is capable of running in real-time. If this is the case, why wasn't this done? With the data presented, the major outcome that should be reported in the abstract is the fully-online performance with notes about how this can be improved offline.

Although the language model is theoretically quite capable of running in real-time, the software engineering and development needed to implement this would have required a large amount of effort that we felt was not germane to the core scientific questions on which we were focused. Although a real-time demonstration of the language model is potentially compelling as a demonstration, we felt that this portion of an eventual clinical system would best be left to experts in language modeling and future work.

Nevertheless, we do agree that the online results should be reported in the abstract and given precedence. We have changed the abstract to read as follows:

> With this BCI, our study participant, ~~(~~whose hand was paralyzed from spinal cord injury,~~)~~ achieved typing speeds that exceed those of any other BCI yet reported: 90 characters per minute at 94.1% raw accuracy online, and >99% accuracy offline with a general-purpose autocorrect.

10) Discussion, line 252: This sentence states that the subject's hand never movement, but a video is shown to highlight the micromotions. Was the subject intending to trace the letter trajectory, even if his injury likely limited his ability to do so accurately?

Yes, the participant was *attempting* to handwrite each letter (see our longer response to this issue below under "H. Clarity and context"). Note that although the participant could generate twitches/micromotions, his hand was severely paralyzed and retained no useful function. We changed this sentence to read:

> To achieve high performance, we developed new decoding methods to overcome two key challenges: (1) lack of observable behavior during long sequences of self-paced training data (our participant's hand ~~never moved~~was paralyzed), and (2) limited amounts of training data.

11) Methods, lines 689-692: Additional detail about the linear transformation and process of fitting separate input layers each day should be stated here, or clearly linked to the supplemental methods. The supplemental figure alone is not sufficient for understanding these steps.

This section now links clearly to the supplemental methods:

> To account for differences in neural activity across days [11,13], we separately transformed each days' neural activity with a linear transformation that was simultaneously optimized with the other RNN parameters (see Supplemental Methods, "Combining Data Across Days" section).

G. References:
References are appropriate. The only comment is with regard to Reference 24 that is cited to show that EEG-BCI has achieved speeds of 60 characters per minute. This is a generous statement and other limitations could be noted given that that level of performance is not typical and was obtained from some healthy subjects due cued typing. This is a minor point.

Thank you, and indeed we wanted to, if anything, err in the direction of being generous. But we agree, and we have updated this discussion section to be more comprehensive. It now reads as follows:

> Commonly used BCIs for restoring communication to people who can't move or speak are either flashing EEG spellers[14–19] or 2D point-and-click computer cursor-based BCIs for selecting letters on a virtual keyboard [20,13,3]. Typical EEG spellers based on P300s or motor imagery achieve 1-5 characters per minute in people with paralysis [14–16,18,19]. EEG spellers that use visually evoked potentials have achieved speeds of 60 characters per minute [17] in able-bodied people, but have important usability limitations, as they tie up the eyes, are not typically self-paced, and require panels of flashing lights on a screen that take up space and may be fatiguing.

H. Clarity and context:
1) In the abstract, results, and discussion, the authors refer to the subject as being completely paralyzed below the neck and that he performed "imagined" hand movements. However, they note that the subject retained some movement of his shoulders and that he had micromotions of his hand during the handwriting task. It would be more appropriate to describe any residual function in the subject's arm and hand. Additionally, the authors should clarify if the subject was imagining the movements or attempting them (resulting in micromotions). See for example previous work from this group: Rastogi, A., Vargas-Irwin, C.E., Willett, F.R. et al. Neural Representation of Observed, Imagined, and Attempted Grasping Force in Motor Cortex of Individuals with Chronic Tetraplegia. Sci Rep 10, 1429 (2020).

Thank you for raising this lack of clarity. Our participant was *attempting* to handwrite each letter, thus resulting in micromotion of the paralyzed hand. Although the participant was imagining that he was holding a pen over a piece of paper, the movement itself is better described as attempted instead of imagined. We chose to instruct attempted movement instead of imagined movement because, as you note, prior work has demonstrated that attempted movement evokes stronger neural activity than purely imagined movement. The following sentence in the first paragraph of the Results describes the movement as attempted:

> We instructed T5 to 'attempt' to write as if his hand was not paralyzed (while imagining that he was holding a pen on a piece of ruled paper).

We have also substituted all instances of the word 'imagined' with 'attempted' throughout the manuscript. In the title, we have simply removed the word 'imagined'. The title now reads: "High-performance brain-to-text communication via handwriting". We felt that including the phrase "attempted handwriting" in the title may confuse readers who are not in the BCI field since, to our knowledge, "attempted movement" is BCI-specific jargon.

2) The abstract should report the typing speeds and accuracy that were achieved completely online without the language model since that is most representative of actual performance. It would be appropriate to also include results with offline enhancements as these would be acceptable in many contexts (such as writing an email).

Thank you for pointing this out, we agree and have changed the abstract to read as follows:

> With this BCI, our study participant, ~~(~~whose hand was paralyzed from spinal cord injury,~~)~~ achieved typing speeds that exceed those of any other BCI yet reported: 90 characters per minute at 94.1% raw accuracy online, and >99% accuracy offline with a general-purpose autocorrect.

We envision that in a final system, a language model could even be integrated into the BCI itself and thus used for all applications (much like speech recognition systems which rely heavily on language modeling). Thus, we think it is important and relevant to report the results with a language model applied.

Point-by-point responses to referee #2

Willett et al. present an intracortical BCI (iBCI) decoding approach for classifying many characters to enable rapid typing. Their approach uses an RNN architecture to perform classification on neural activity as the subject imagines writing letters/words/sentences. They achieve typing speeds up to 90 characters per minute with above 94.5% accuracy in one subject, which significantly outperforms previous communication iBCIs. They demonstrate the system works across several sessions and both for copying text and free expression. The authors further provide analyses to provide intuition for why their approach succeeds--they achieve high classification accuracy by having the user perform a task that generates highly discriminable neural activity.

Overall, the manuscript is very well written and represents a clear and important advance in the field of BCIs. The technical innovations of the paper include 1) methods for creating training datasets when there is minimal available information (since the subject imagined moving) and 2) methods for leveraging the power of RNNs even with relatively limited data. The approaches for challenge 2 primarily use techniques common in ANNs (data augmentation) and techniques previously shown to be useful in animal studies (adding external noise to increase robustness of the networks). The solutions to challenge 1 appear relatively novel, and are certainly new to the field of BCIs. The approach/conceptual innovation of the paper is a shift away from decoding continuous control towards a method that provides accurate classification even for a relatively large 31 character set. To my knowledge this is a notable departure from prior work.

We are gratified that the reviewer expresses that this is an important advance for BCIs, and that the training methods and approach is genuinely novel. We thank the reviewer for their thorough read of the manuscript and their insightful and helpful suggestions.

My primary concern with the manuscript is how the author's frame the work's overall approach which should more clearly emphasize the shift towards classification. As their work demonstrates, this shift can be powerful but it is also very specialized to this task. The manuscript's current comparisons to previous state-of-the-art (Pandarinath et al.) and figure 3 fail to fully make the distinction between continuous decoding of a cursor for selecting keys on a keyboard from their BCI performing a 31-way classification. Figure 3, for instance, almost implies that Pandarinath and prior BCIs were trying to classify straight line movements, which they did not. The authors' point that discriminability of the neural activity patterns directly impacts classifier performance is well taken. And provides an intuition for why having users imagine writing letters enabled their advance. But the manuscript needs to be very clear that in and of itself does not explain why they achieve higher performance. It explains why they were able to classify a large alphabet successfully for the first time. They then achieve higher performance compared to prior work because their classifier can predict letters more quickly than the average translation + click time of continuous control cursor tasks. The primary reason I emphasize this distinction is that their classification approach solves the problem of typing quite well, but does not provide a mechanism that necessarily generalizes to other tasks that are more continuous in nature like controlling a robotic limb (the authors do not claim this, but I think it's important the paper itself makes this distinction more clearly).

Thank you for pointing out this potential point of confusion. Indeed, the idea of improving classification performance by increasing the temporal dimensionality of the decoded movements is specific to discrete BCIs (and thus does not apply to BCIs that restore continuous motion). Additionally, your point is well taken that the increased decodability of handwritten letters is not necessarily the only reason why the handwriting BCI was able to go twice as fast as a point-and-click BCI. However, we theorize that it is *one* important factor that enabled the speed increase.

Fundamental to our argument is the idea that the speed of a point-and-click BCI is limited primarily by decoding accuracy. During parameter optimization of point-and-click BCIs, the cursor gain (speed scaling parameter) is typically increased as much as possible to increase typing speed, until it reaches a point where the cursor becomes uncontrollable due to decoding errors that push the cursor around randomly [1]. Thus, we do believe it is important to ask: how is it that our handwriting BCI was able to achieve similar levels of decoding accuracy (mid-90s) while going twice as fast? In other words, why couldn't the

point-and-click BCI go twice as fast as it did? Why did accurate point-and-click movement become essentially impossible at only half the speed of the handwriting BCI?

Our explanation is that different handwritten characters are easier to tell apart from each other than different straight-line movements. It is important to note that there is still a large gap between the performance of continuous cursor BCIs and able-bodied movement, suggesting that point-and-click BCIs are still limited primarily by decoding accuracy and not by fundamental behavior limits. Consistent with this idea, data on 1-finger typing on smartphones shows that the average typing rates are much higher than what was achieved in (our) Pandarinath et al. *eLife* 2017 publication (40 characters per minute vs. 120+) [2], further suggesting that point-and-click BCI speeds are not limited by fundamental brain/behavior limits on point-to-point movement and click/selection. Finally, we note that letters have more movement segments than a straight-line movement does (several letters have multiple straight lines in them). Despite this, handwriting movements could be decoded at greater speeds than point-to-point movements, which also suggests that point-and-click BCI speeds have not yet approached the fundamental limit of human behavior.

[1] Willett, Francis R., Brian A. Murphy, William D. Memberg, Christine H. Blabe, Chethan Pandarinath, Benjamin L. Walter, Jennifer A. Sweet, et al. "Signal-Independent Noise in Intracortical Brain–Computer Interfaces Causes Movement Time Properties Inconsistent with Fitts' Law." *Journal of Neural Engineering* 14, no. 2 (2017): 026010. https://doi.org/10.1088/1741-2552/aa5990.

[2] Palin, Kseniia, Anna Maria Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. "How Do People Type on Mobile Devices? Observations from a Study with 37,000 Volunteers." In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*, 1–12. MobileHCI '19. Taipei, Taiwan: Association for Computing Machinery, 2019. https://doi.org/10.1145/3338286.3340120.

To better explain this point, we added additional text to the motivating paragraph of the Results section, which now reads as follows:

> To our knowledge, 90 characters per minute is the highest typing rate yet reported for any type of BCI (see Discussion). For intracortical BCIs, the highest performing method has been point-and-click typing with a 2D computer cursor, peaking at 40 characters per minute [7] (this record was also set by participant T5 3 years earlier; see SVideo 3 for a direct comparison). The speed of point-and-click BCIs is limited primarily by decoding accuracy. During parameter optimization, the cursor gain is increased so as to increase typing rate, until the cursor moves so quickly that it becomes uncontrollable due to decoding errors[20]. How is it then that handwriting movements could be decoded more than twice as fast, with similar levels of accuracy?

Importantly, we also now clarify that there are other factors to consider when comparing the handwriting BCI to a point-and-click BCI:

> These results suggest that time-varying patterns of movement, such as handwritten letters, are fundamentally easier to decode than point-to-point movements, and can thus enable higher communication rates (although other important differences between continuous point-and-click BCIs and discrete handwriting BCIs, such as the time taken to execute a click, also contribute to their speed difference).

Additionally, we now explicitly clarify that the concept of increasing the temporal dimensionality of the decoded movements can be applied to improve *discrete* BCIs only:

> The concept of intentionally increasing temporal dimensionality could be applied more generally to improve any discrete (but not continuous) BCI that enables ~~discrete~~ selection between a set of options, ~~(~~by associating these options with time-varying gestures as opposed to simple movements~~)~~.

Finally, we now draw a clearer distinction between this work and prior work on discrete intracortical BCIs:

> Prior discrete BCIs have typically used simple directional movements as opposed to spatiotemporally patterned movement, which may have limited their accuracy and/or the size of the movement set[22,23].

[22]Musallam S, Corneil BD, Greger B, Scherberger H, Andersen RA (2004) Cognitive Control Signalsfor Neural Prosthetics. Science 305.

[23]Santhanam G, Ryu SI, Yu BM, Afshar A, Shenoy KV (2006) A high-performance brain–computer interface. Nature 442:195–198.

Specific points:
Is this the same T5 patient from Pandarinath et al. 2017? If so, it would strengthen the manuscript's claims to highlight this direct comparison (where they are also potentially at a disadvantage if studies were performed later with likely lower quality neural recordings).

Yes, it is the same participant. We now highlight this fact explicitly:

> For intracortical BCIs, the highest performing method has been point-and-click typing with a 2D computer cursor, peaking at 40 characters per minute [3] (this record was also set by participant T5 three years earlier; see SVideo 3 for a direct comparison).

If this is the same patient T5, the manuscript should mention that this subject did have the best performance of the 3-subject cohort in that prior study. While the performance advantages of their decoder are clear, given the single subject demonstration this potential subject-to-subject variability should be discussed.

We now highlight more explicitly in the Discussion that this study was from a single participant, and highlight the potential variability across participants:

> Finally, it is important to recognize that ~~our~~ the current system is a proof-of-concept that a high-performance handwriting BCI is possible (in a single participant capable of handwriting prior to his injury); it is not yet a complete, clinically viable system. More work is needed to demonstrate high performance in additional people, expand the character set (e.g. capital letters), enable text editing and deletion, and maintain robustness to changes in neural activity without interrupting the user for decoder retraining. More broadly, intracortical microelectrode array technology is still maturing, and requires further demonstrations of longevity, safety, and efficacy before widespread clinical adoption[33,34]. Variability in performance across participants is also a potential concern that may require improvements in intracortical recording technology to increase consistency (in a prior study, T5 achieved the highest performance of 3 participants[7]).

The increase in characters per minute (Figure 2C) should be discussed. In addition to being more accurate over time (which may be attributed to the addition of previous day's data to the RNN training dataset), there is also an observed increase in typing speed (characters per minute). Is this also due to additional training data or other phenomena? A retrospective analysis with decoder performance on a single day's data would be useful information.

We believe that the increase in speed over time is due to T5 becoming more comfortable with writing quickly. Our handwriting BCI is different from a point-and-click BCI in that there is no gain (speed scaling) parameter that effectively determines the typing rate of the BCI. Instead, the pace is set entirely by the user, who chooses how fast to write each letter (i.e. the BCI does not constrain the user to write more slowly in order to maintain accuracy – the writing speed is entirely up to the user). Although we always instructed T5 to write as quickly as possible, he reported increasing his speed over time as he began to trust that the BCI would maintain high accuracies at faster speeds. We note that there is no way to know, objectively, why T5 decided to increase his writing speed other than the reasoning that he reports to us. We therefore added the following sentence to the Results:

> T5 reported increasing his writing speed over time as he gained confidence that the BCI could maintain its accuracy at high speeds.

Note that although characters per minute increased over time, the accuracy did not. Instead, there was a modest increase in *error rate* over time (we observed an average error rate of 4.3% on the first day and 5.4% on the last day). For convenience, Fig. 2C is reproduced below, which shows a small increase in error over time which is interrupted by trial day 1237 (which appears to be an outlier).

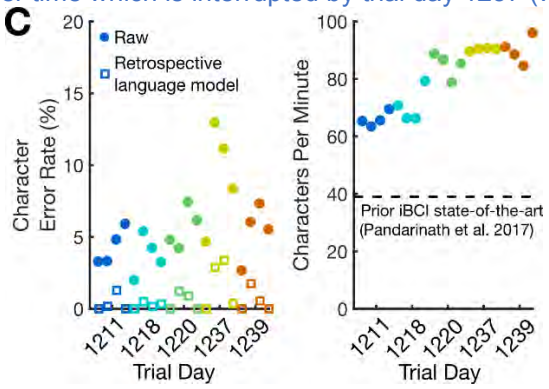
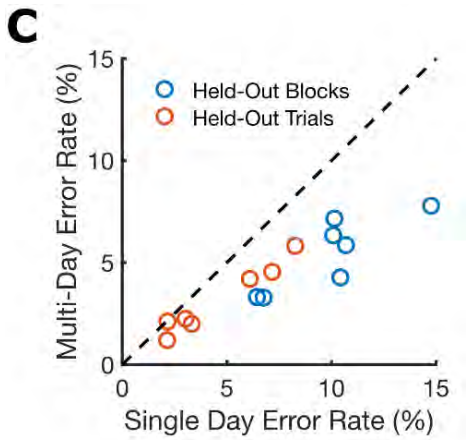


Fig. 2C. Error rates (edit distances) and typing speeds are shown for five days, with four blocks of 7-10 sentences each (each block indicated with a single circle).

We speculate that it is easier to classify at slower speeds because of an increased amount of neural data per character that can be used to decide that character's identity, effectively increasing the overall SNR of the data available to the decoder.

As suggested by the reviewer, it is indeed the case that adding more training data increases the accuracy of the RNN, as shown by Supplementary Figure 2C (reproduced below). This figure panel compares the offline decoding performance of an RNN trained on all days ("Multi-Day Error Rate") to the offline decoding performance of separate RNNs trained on each day alone ("Single Day Error Rate"). Training on all days relative to just a single day reduced the error rate percentage by 4.7 (95% CI = [4.1, 5.3]).



Supp Fig. 2C. Training an RNN with all of the datasets combined improves performance relative to training on each day separately. Each circle shows the performance on one of seven days.

The experimental setup for real-time decoding should be clarified. Did the subject see the raw outputs during the task?

Yes, T5 saw the raw outputs (i.e., each letter, whether correct or incorrect) appear on the screen during the task. In the Results section, we offer the following description:

> T5 copied each sentence from an onscreen prompt, attempting to handwrite it letter by letter, while the decoded characters appeared on the screen in real-time as they were detected by the RNN (SVideo 1, Table S2).

To make this clearer, we amended the legend of Figure 2 to now state the following:

> Finally, the character probabilities were thresholded to produce "Raw Output" for real-time use (when the "new character" probability crossed a threshold at time *t*, the most likely character at time *t*+0.3s was emitted from the decoder and shown on the screen).

The authors nicely isolate the effect of the RNN from the more discriminable neural activity (supplemental table S4). Though I think they somewhat overstate the importance of the RNN compared to HMM in the main manuscript methods, since the RNN's main advantage is it's robustness against noise (by the authors design with noise-training for the RNN). It's actually quite noteworthy that the neural activity differences alone still lead to solid performance in a 31-way classification task with a linear HMM.

Thank you, we too agree that it is very encouraging that a simpler decoding algorithm was able to achieve good performance. Nevertheless, we do think that the numbers in table S4 (0.23% error rate with an RNN and 2.96% with an HMM, when a language model was applied to both) actually do show a large improvement for the RNN, when one considers that a 0.23% error rate means *ten times* fewer errors.

We made the following change to the Methods section where this issue is discussed to make our statement more quantitatively precise:

> We found that a recurrent neural network decoder ~~strongly~~ outperformed a simple hidden Markov model decoder (Table S4, 0.23% error rate vs. 2.93% error rate under the most favorable conditions for the HMM, and 0.70% vs. 80.1% under the least favorable), but note that quite-discriminable neural activity enabled even the HMM decoder to perform reasonably well.

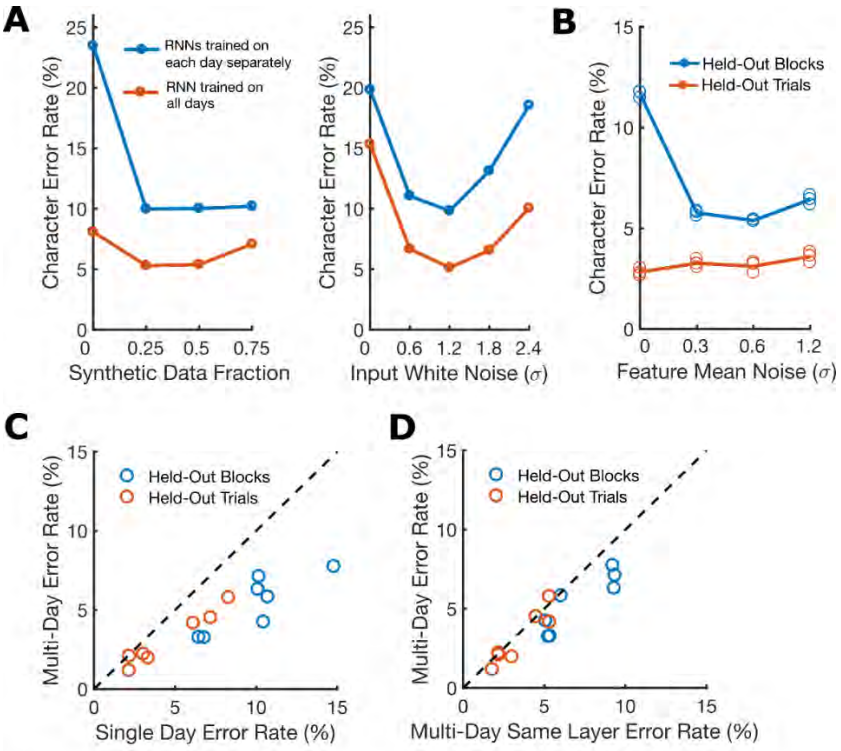
The "character stretch factor" is not well explained in the supplements. What does this factor represent?

Thank you for pointing this out. We added the following sentence of explanation to the supplement:

> The stretch factor determines how the character template is contracted or dilated in time (using linear interpolation) to be longer or shorter than its average duration.

Figure S3C and D -- are these differences statistically significant? More quantification rather than just "substantially improved" would be useful.

Thank you for this suggestion. We added a table to Figure S3 which summarizes the error rate improvement generated by each RNN parameter/technique, with a 95% confidence interval so that statistical significance can be assessed. The new figure is reproduced below (the new table is shown in panel E):



E

Parameter	Panel	Comparison	Error Rate Improvement [95% CI]
Synthetic data fraction	A (left)	Single-day RNN (0 vs. 0.25)	12.9 [11.9, 14.0]
		Multi-day RNN (0 vs. 0.25)	2.7 [2.2, 3.2]
Input white noise	A (right)	Single-day RNN (0 vs. 1.2)	9.8 [8.7, 11.0]
		Multi-day RNN (0 vs. 1.2)	10.1 [9.2, 11.1]
Feature mean noise	B	Held-out blocks (0 vs. 0.3)	5.7 [4.9, 6.5]
		Held-out trials (0 vs. 0.3)	-0.7 [-1.3, -0.1]
Single-day vs. multi-day	C	Held-out blocks	4.7 [4.1, 5.3]
		Held-out trials	1.5 [0.6, 2.4]
Day-specific layers vs. same layer	D	Held-out blocks	1.6 [1.1, 2.7]
		Held-out trials	0.1 [-0.7, 0.7]

Additionally, we updated the Methods text to include quantifications of the performance improvement.

> Including multiple days of data, and fitting separate input layers for each day, ~~substantially~~ significantly improved performance (decreased the error rate percentage by 4.7 and 1.6, respectively; ~~(~~SFig. 3C-D).

I'm left with an impression that many design choices in the machine learning algorithms were hand tailored. This is fine, especially for initial proof of concept. But the discussion might benefit from mentioning that methods for more automated algorithm development/training will be needed for wider utility.

Indeed, many of the parameters were hand-tuned (as we think is typical in machine learning applications). For later days, some hyperparameters were tuned via a random search over possible parameter values. We added the following text to the Methods section to highlight this issue:

> Hyperparameter values were largely hand-tuned; for later sessions, some parameters were tuned via small random searches over possible parameter values (see Supplemental Methods for values). Ultimately, automated parameter tuning may be required (and would certainly be useful) when applying these techniques to new participants in future clinical applications.

Point-by-point responses to referee #3

A. Summary of the key results
The work reports a single subject's performance using an intracortical BCI that can decode imagined handwriting movements from neural activity in motor cortex and map it to text in real-time. Overall the work fits within the growing body of literature intended to demonstrate faster and more accurate BCIs with improved understanding of movement encoding and more sophisticated decoding methods.
Outstanding features of the work are:
• Typing speeds of on-screen prompt at 90 characters per minute at 99% error rate with the use of a general-purpose autocorrect and 73.8 characters at 8.54% error rate for self-generated sentences (2.25% with a language model) are significant advances over the highest reported point and-click typing with a 2D computer cursor, peaking at 40 characters per minute. Results open a new approach for BCIs and demonstrate the feasibility of accurately decoding imagined handwriting movements years after losing ability to move and speak.
• The combination of probabilistic and modeling frameworks forming a hierarchical decoding approach with multiple time scales to combat neural signal variability.
• An interesting theoretical principle is proposed in which point-to-point movements may be harder to decode from one another compared to handwritten letters. Authors attribute this to the idea that temporally complex movements, such as handwriting, may be fundamentally easier to decode than point-to-point movements.

We are gratified that the reviewer expresses that this is a new approach that makes significant gains in BCI performance. We thank the reviewer for their thorough read of the manuscript and their insightful and helpful suggestions.

B. Originality and significance:

The paper draws upon handwriting or touch typing as a faster means to communicate by a specific population of neurologically impaired subjects. The work is an extension to this group's past contributions on BCIs for communications to the 'locked-in' population. Results presented here would be of interest to people in the BCI community who are working on restoring communication to these people who cannot move or speak.

Overall, the work is significant and original but can be better articulated.

We thank the reviewer for noting the originality and significance of the work, and for their detailed suggestions below on how to improve its presentation. We have made every attempt to follow these helpful recommendations, and we believe that the manuscript is much stronger as a result. Again, thank you.

First, authors should cite the prevalence of such conditions to put this contribution in the right context.

Thank you for this suggestion. We added the following to the Discussion:

> Locked-in syndrome (paralysis of nearly all voluntary muscles) severely impairs or prevents communication, and is most frequently caused by brainstem stroke or late-stage ALS (estimated prevalence of locked-in syndrome: 1 in 100,000 [25]).

[25]Pels, Elmar G.M., Erik J. Aarnoutse, Nick F. Ramsey, and Mariska J. Vansteensel. "Estimated Prevalence of the Target Population for Brain-Computer Interface Neurotechnology in the Netherlands." *Neurorehabilitation and Neural Repair* 31, no. 7 (July 2017): 677–85. https://doi.org/10.1177/1545968317714577.

Second, the primary performance metric is typing speed. However, on numerous occasions, the authors attempt to give the impression that this is the primary metric that could be the sole determinant for adopting the technology. While this metric is undoubtedly critical, I think the authors should reframe this argument differently, in that it is the combination of a number of factors —one of which is typing speed—

that would ultimately make the technology a first choice for the intended population. For example, the recalibration of decoders is another such factor, and while it is acknowledged by the authors that their approach is quite extensive, it is unclear how much time and resources the recalibration process takes (see detailed comments below). Another factor is the integrity of the signals over the longevity of the implant, which is a prime issue with all invasive technology (see detailed comments below).

Thank you for the important recommendation to reframe the argument differently, including the suggestion to address the decoder calibration process and electrode array longevity, which has led to new analyses and discussion points (described below) which we believe have significantly improved the manuscript. Since these are important themes that recur in this (R3's) review, we take some space here to outline our overall philosophy and approach, as well as highlight each major addition to the paper.

Ultimately, we see this study as being primarily focused on demonstrating the feasibility of decoding handwriting movements well enough to substantially increase BCI communication rates. This opens the door to a promising new approach for iBCIs, which we believe is an important and exciting advance. To our knowledge, this is the first demonstration that rapid sequences of dexterous movements can be decoded in a person who has been paralyzed for several years. However, by no means does our BCI yet constitute a 'complete product' that would be appropriate for immediate clinical adoption.

First, as the reviewer has noted here and below, array longevity is a critical issue for any intracortical BCI. Before a product is taken to market, a systematic study must be conducted which demonstrates longevity across many subjects. While no such study has yet been published, preliminary results from several studies indicate that arrays retain their functionality for several years in people, with multiple examples of retained functionality for 1000+ days (Bullard et al., 2020; Simeral et al., 2011). In this current study, high performance was obtained 1200+ days post-implant, and these arrays continue to record high-quality spiking activity (see below). We are currently preparing a separate manuscript summarizing array safety and longevity data from all 14 participants in the BrainGate pilot clinical trials (collected over a span of 15 years), which will be the first systematic study of its kind in people. Given the complex and multiple factors contributing to array longevity, we believe this important fundamental question is outside the scope of the current work (beyond simply noting that the results were obtained 3+ years post-implant and that the arrays continue to record high-quality signals). We now clearly highlight this issue in the Discussion and have added a new supplementary figure demonstrating that the arrays continue to record high-quality spiking activity.

Second, as the reviewer notes, minimizing decoder recalibration time is also an important problem for intracortical BCIs (as well as many non-invasive BCIs). This issue must also be addressed before a viable product can be taken to market. However, we see this as another deep topic in and of itself, which has been the sole focus of several recent studies (Jarosiewicz et al., 2015; Dyer et al., 2017; Degenhart et al., 2020). For example, one new method uses an unsupervised approach to track a stable subspace of neural activity over time (Degenhart et al., 2020); the evaluation and design of this method was the subject of an entire paper. Additionally, to our knowledge, daily decoder recalibration is still standard practice in the intracortical BCI field and many important papers have used this method [e.g. (Hochberg et al., 2006, 2012; Collinger et al., 2013; Bouton et al., 2016; Ajiboye et al., 2017; Pandarinath et al. 2017; Nuyujukian et al. 2018]). We think it is therefore reasonable to leave this aspect of handwriting decoding to be more fully investigated in future work. Only now that we have shown that handwriting decoding can achieve higher performance than any other communication BCI, is it properly motivated to begin searching for algorithms that can minimize the calibration data needed to retrain a handwriting decoder.

Nevertheless, we wholeheartedly agree that it is important to both (1) clearly highlight the issue in the Results and Discussion, and (2) preliminarily address whatever key issues we can while remaining within the scope of this work (which we have done via new data analyses, presented below and in the paper). We have taken the following four actions to address the longevity and recalibration issues.

(1) We added new Discussion paragraphs which more clearly address the limitations of the current work, including limitations with intracortical array technology in general (e.g., that more studies are needed to show array longevity). We also give some broader context as to why we believe intracortical technology is

a promising route forward for restoring rapid communication, despite not necessarily being ready for widespread clinical adoption at the current time. For convenience, we reproduce these new paragraphs here:

> Finally, it is important to recognize that ~~our~~ the current system is a proof-of-concept that a high-performance handwriting BCI is possible (in a single participant capable of handwriting prior to his injury); it is not yet a complete, clinically viable system. More work is needed to demonstrate high performance in additional people, expand the character set (e.g. capital letters), enable text editing and deletion, and maintain robustness to changes in neural activity without interrupting the user for decoder retraining. More broadly, intracortical microelectrode array technology is still maturing, and requires further demonstrations of longevity, safety, and efficacy before widespread clinical adoption[33,34]. Variability in performance across participants is also a potential concern that may require improvements in intracortical recording technology to increase consistency (in a prior study, T5 achieved the highest performance of 3 participants[7]).
>
> Nevertheless, we believe the future of intracortical BCIs is bright. Current microelectrode array technology has been shown to retain functionality for 1000+ days post implant[35,36] (including in this work - see SFig 6 for examples of high-quality spiking activity), and has enabled the highest BCI performance to date compared to other recording technologies (EEG, ECoG) for restoring communication[7], arm control[2,5], and general-purpose computer use[37]. New developments are underway for implant designs that increase the electrode count by an order of magnitude, which will further improve performance and longevity[33,34,38,39]. Finally, we envision that a combination of algorithmic innovations and improvements to device stability will continue to increase the robustness of intracortical BCIs, which have so far typically required daily decoder retraining to account for changes in neural recordings that accrue over time (although see e.g. [40,41]). In this study, offline analyses showed that large amounts of daily calibration data are not needed to achieve good performance, and that an unsupervised approach is promising for enabling behind-the-scenes decoder retraining without interrupting the user. Other recent work has also advanced new algorithms for unsupervised decoder retraining[42,43], making important steps towards robust, easy-to-use intracortical BCI systems.

(2) We added a new figure (Fig. 3) to the main text focused solely on decoder recalibration. This figure reports results from new offline analyses that quantify how much calibration data is needed to achieve high performance. As noted by the reviewer, our original study design used a large amount of calibration data to retrain the decoder each day (50 sentences). However, this was not because it was *necessary* to use that many sentences to achieve good performance. Our new analysis demonstrates that high performance could have be obtained with much less data (10 sentences). We also assess whether the amount of time that passes between sessions affects how much calibration data is needed. We show that, when 7 days or less pass between sessions, it is possible to achieve good performance even with *no* decoder calibration. Moreover, we demonstrate that an unsupervised decoder recalibration method can achieve high performance without requiring any explicit data labels. This is promising from a clinical viability standpoint, as it suggests that a decoder recalibration routine may be able to run in the background without interrupting the user. We believe that these new results improve the paper significantly by (a) highlighting this important issue and (b) showing initial promise that it is possible to achieve high performance with modest amounts of calibration data.

We reproduce the new figure (Fig. 3) and accompanying Results text below for convenience:

> Following standard practice for BCIs (e.g. [1,2,19,4,5]), we retrained our handwriting decoder each day before evaluating it, with the help of "calibration" data collected at the beginning of each day. Retraining helps account for changes in neural recordings that accrue over time. Ideally, to reduce the burden on the user, little or no calibration data would be required. In a retrospective analysis of the copy typing data reported above in Fig. 2, we assessed whether high performance could still have been achieved using less than the original 50 calibration sentences per day (Fig. 3A). We found that 10 sentences (8.7 minutes) were enough

to achieve a raw error rate of 8.5% (1.7% with a language model), although 30 sentences (26.1 minutes) were needed to match the raw online error rate of 5.9%.

However, our copy typing data were collected over a 28-day time span, possibly allowing larger changes in neural activity to accumulate. We therefore tested whether more closely-spaced sessions reduce the need for calibration data (Fig. 3B), using an offline analysis of copy typing data across 8 sessions. We found that when only 2-7 days passed between sessions, performance was reasonable with *no* decoder retraining (11.1% raw error rate, 1.5% with a language model). Finally, we tested whether decoders could be retrained in an unsupervised manner by using a language model to error-correct and retrain the decoder, thus bypassing the need to interrupt the user for calibration (i.e. by recalibrating automatically during normal use). Encouragingly, unsupervised retraining achieved a 7.3% raw error rate (0.84% with a language model) when sessions were separated by 7 days or less (see Methods & Supplemental Methods for details). Ultimately, whether decoders can be successfully retrained with minimal recalibration data depends on how quickly the neural activity changes over time. We assessed the stability of the neural patterns associated with each character and found high short-term stability (mean correlation = 0.85 when 7 days apart or less), and neural changes that seemed to accumulate at a steady and predictable rate (SFig. 4 provides a quantitative visualization). The above results are promising for clinical viability, as they suggest that unsupervised decoder retraining, combined with more limited supervised retraining after longer periods of inactivity, may be sufficient to achieve high performance.

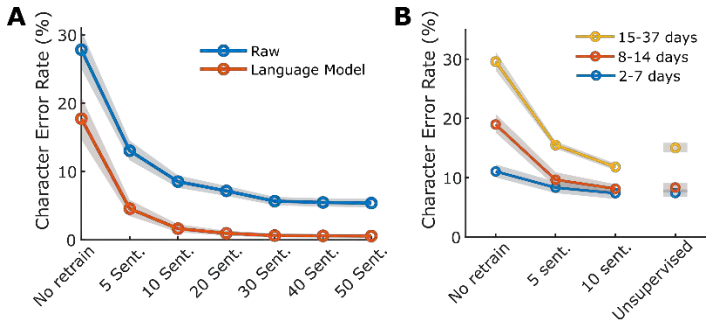
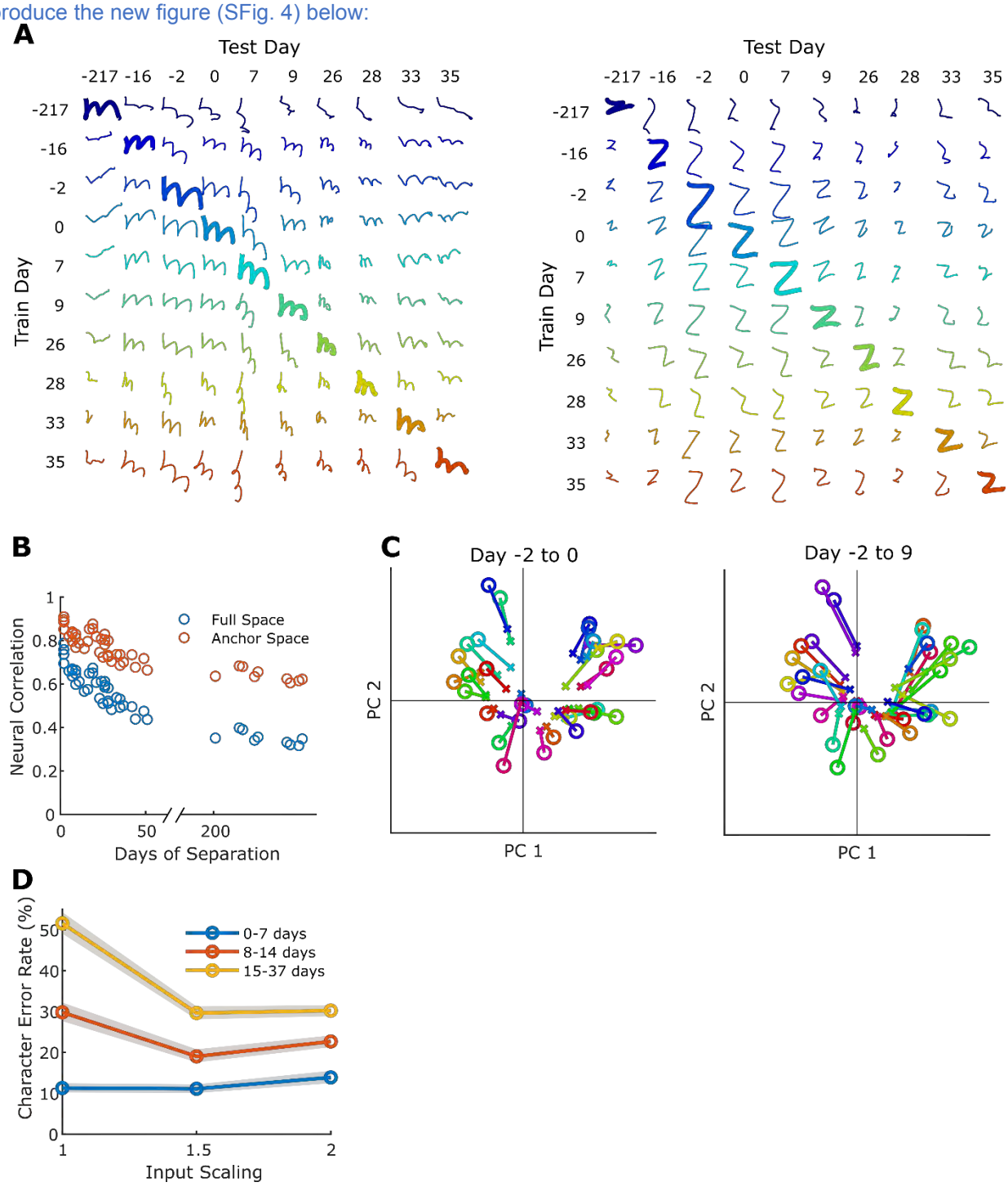


Figure 3. Performance remains high when decoder retraining is limited or omitted. (A) To account for neural activity changes that accrue over time, we retrained our handwriting decoder each day before evaluating it. Here, we simulate offline what the decoding performance shown in Fig. 2 would have been if less than 50 calibration sentences were used. Lines show the mean error rate across all data and shaded regions indicate 95% CIs (computed via bootstrap resampling of single trials, N=10,000). **(B)** Copy typing data from eight sessions were used to assess whether less calibration data are required if sessions occur closer in time. All session pairs (X, Y) were considered. Decoders were first initialized using training data from session X and earlier, and then evaluated on session Y under different retraining methods (no retraining, retraining with limited calibration data, or unsupervised retraining). The average raw character error rate is plotted for each category of time elapsed between sessions X and Y, and for each retraining method. Shaded regions indicate 95% CIs.

(3) We added a new supplemental figure (now SFig. 4) that assess the stability of the neural patterns associated with each character over time, since this is a critical issue that ultimately determines how much data is needed for daily decoder recalibration. We found high short-term stability (mean correlation = 0.85 when 7 days apart or less), and neural changes that seemed to accumulate at a steady and predictable rate. Again, this is promising for the possibility of recalibrating decoders with limited amounts of data (or even in an unsupervised manner without interrupting the user). We also found that as neural activity slowly rotates into new neural subspaces over time, it tends to shrink towards the origin in the original neural subspace, but otherwise retains a very similar structure there. This suggests the following idea: if we scale up the inputs to the decoder when transferring it to a new day, this might prevent the decoder from perceiving smaller-than-expected modulation in the original subspace. We found that input re-scaling does indeed improve performance, and we include this result as part of the supplemental figure. We think this is a useful principle that could benefit other types of BCIs as well.

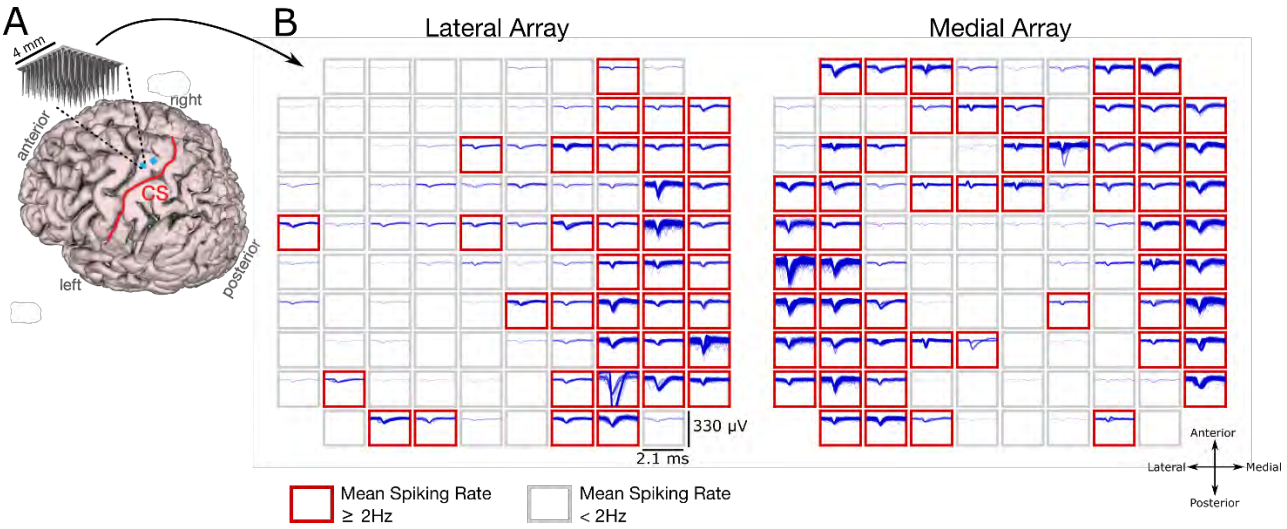
We reproduce the new figure (SFig. 4) below:



Supplemental Figure 4. Changes in neural recordings across days. (A) To visualize how much the neural recordings changed across time, decoded pen tip trajectories were plotted for two example letters ("m" and "z") for all ten days of data (columns), using decoders trained on all other days (rows). Each session is labeled according to the number of days passed relative to Dec. 9, 2019 (day #4). Results show that although neural activity patterns clearly change over time, their essential structure is largely conserved (since decoders trained on past days transfer readily to future days). (B) The correlation (Pearson's *r*)

between each session's neural activity patterns was computed for each pair of sessions and plotted as a function of the number of days separating each pair. The *r* values were computed by correlating the spatiotemporal neural patterns of average firing rates associated with each character (see Supplemental Methods for more detail). Blue circles show the correlation computed in the full neural space (all 192 electrodes) while red circles show the correlation in the "anchor" space (top 10 principal components of the earlier session). High values indicate a high similarity in how characters are neurally encoded across days. The fact that correlations are higher in the anchor space suggests that the structure of the neural patterns stays largely the same as it slowly rotates into a new space, causing shrinkage in the original space but little change in structure. (C) A visualization of how each character's neural representation changes over time, as viewed through the top two PCs of the original "anchor" space. Each "o" represents the neural activity pattern for a single character, and each "x" shows that same character on a later day (lines connect matching characters). The left panel shows a pair of sessions with only two days between them ("Day -2 to 0"), while the right panel shows a pair of sessions with 11 days between them ("Day -2 to 9"). The relative positioning of the neural patterns remains similar across days, but most conditions shrink noticeably towards the origin. This is consistent with the neural representations slowly rotating out of the original space into a new space, and suggests that scaling-up the input features may help a decoder to transfer more accurately to a future session (by counteracting this shrinkage effect). (D) Similar to Fig. 3B, copy typing data from eight sessions was used to assess offline whether scaling-up the decoder inputs improves performance when evaluating the decoder on a future session (when *no* decoder retraining is employed). All session pairs (X, Y) were considered. Decoders were first initialized using all data from session X and earlier, then evaluated on session Y under different input scaling factors (e.g., an input scale of 1.5 means that input features were scaled up by 50%). The average raw character error rate is plotted for each category of time elapsed (between sessions X and Y) and each retraining method. Shaded regions indicate 95% CIs. Results show that when long periods of time pass between sessions, input-scaling improves performance. We therefore used an input scaling factor of 1.5 when assessing decoder performance in the "no retraining" conditions of Fig. 3.

(4) We added a new supplemental figure (now SFig. 6) to demonstrate that high quality spiking activity can still be recorded on many of the microelectrodes 3+ years post-implant. This demonstrates that intracortical microelectrode arrays have the potential to last for several years in people (although as stated above, additional evidence from more subjects will ultimately be required to systematically demonstrate longevity). We also quantified how many of the total 192 electrodes could still record high-quality spiking activity and now report this number in the Methods (81.9 ± 5.6), which we believe gives the reader useful additional context. We used a simple, conservative metric to estimate if an electrode still recorded spike-like activity that could have arisen from single neurons. Specifically, if the voltage crossed a -4.5 RMS threshold more than 2 times per second on average, the electrode was considered to record spiking activity. Note that a -4.5 RMS threshold excludes almost all background noise (and many electrodes therefore record almost no spiking events at this threshold). Although we could have also spike-sorted these waveforms, spike-sorting is a subjective and somewhat arbitrary process since it is not always clear whether a cluster of waveforms truly belongs to one (and only one) neuron. Thus, this metric is a lower bound on the number of spike clusters (since the activity on each spiking electrode could be sorted into *at least* one cluster). This new figure is reproduced below:



Supplemental Figure 6. Example spiking activity recorded from each microelectrode array. (A) Participant T5's MRI-derived brain anatomy. Microelectrode array locations (blue squares) were determined by co-registration of postoperative CT images with preoperative MRI images. (B) Example spike waveforms detected during a ten second time window are plotted for each electrode (data were recorded on post-implant day 1218). Each rectangular panel corresponds to a single electrode and each blue trace is a single spike waveform (2.1 millisecond duration). Spiking events were detected using a -4.5 RMS threshold, thereby excluding almost all background activity. Electrodes with a mean threshold crossing rate ≥ 2 Hz were considered to have 'spiking activity' and are outlined in red (note that this is a conservative estimate that is meant to include only spiking activity that could be from single neurons, as opposed to multiunit 'hash'). Results show that many electrodes still record large spiking waveforms that are well above the noise floor (the y-axis of each panel spans 330 µV, while the background activity has an average RMS value of only 6.4 µV). On this day, 92 electrodes out of 192 had a threshold crossing rate ≥ 2 Hz.

Taken together, we believe that these new results and discussion points improve the manuscript considerably by providing more perspective about the limitations and potential benefits of intracortical BCIs, while also offering new evidence that minimizing (or in some cases, even eliminating) supervised decoder recalibration appears feasible.

References for this section:

Ajiboye, A.B., Willett, F.R., Young, D.R., Memberg, W.D., Murphy, B.A., Miller, J.P., Walter, B.L., Sweet, J.A., Hoyen, H.A., Keith, M.W., Peckham, P.H., Simeral, J.D., Donoghue, J.P., Hochberg, L.R., Kirsch, R.F., 2017. Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration. The Lancet 389, 1821–1830. https://doi.org/10.1016/S0140-6736(17)30601-3

Bouton, C.E., Shaikhouni, A., Annetta, N.V., Bockbrader, M.A., Friedenberg, D.A., Nielson, D.M., Sharma, G., Sederberg, P.B., Glenn, B.C., Mysiw, W.J., Morgan, A.G., Deogaonkar, M., Rezai, A.R., 2016. Restoring cortical control of functional movement in a human with quadriplegia. Nature 533, 247–250. https://doi.org/10.1038/nature17435

Bullard, A.J., Hutchison, B.C., Lee, J., Chestek, C.A., Patil, P.G., 2020. Estimating Risk for Future Intracranial, Fully Implanted, Modular Neuroprosthetic Systems: A Systematic Review of Hardware Complications in Clinical Deep Brain Stimulation and Experimental Human Intracortical Arrays. Neuromodulation Technol. Neural Interface 23, 411–426. https://doi.org/10.1111/ner.13069

Collinger, J.L., Wodlinger, B., Downey, J.E., Wang, W., Tyler-Kabara, E.C., Weber, D.J., McMorland, A.J., Velliste, M., Boninger, M.L., Schwartz, A.B., 2013. High-performance neuroprosthetic control by an individual with tetraplegia. The Lancet 381, 557–564. https://doi.org/10.1016/S0140-6736(12)61816-9

Degenhart, A.D., Bishop, W.E., Oby, E.R., Tyler-Kabara, E.C., Chase, S.M., Batista, A.P., Yu, B.M., 2020. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. Nat. Biomed. Eng. 1–14. https://doi.org/10.1038/s41551-020-0542-9

Dyer, E.L., Gheshlaghi Azar, M., Perich, M.G., Fernandes, H.L., Naufel, S., Miller, L.E., Körding, K.P., 2017. A cryptography-based approach for movement decoding. Nat. Biomed. Eng. 1, 967–976. https://doi.org/10.1038/s41551-017-0169-7

Hochberg, L.R., Bacher, D., Jarosiewicz, B., Masse, N.Y., Simeral, J.D., Vogel, J., Haddadin, S., Liu, J., Cash, S.S., Smagt, P. van der, Donoghue, J.P., 2012. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. Nature 485, 372–375. https://doi.org/10.1038/nature11076

Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Chen, D., Penn, R.D., Donoghue, J.P., 2006. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature 442, 164–171. https://doi.org/10.1038/nature04970

Jarosiewicz, B., Sarma, A.A., Bacher, D., Masse, N.Y., Simeral, J.D., Sorice, B., Oakley, E.M., Blabe, C., Pandarinath, C., Gilja, V., Cash, S.S., Eskandar, E.N., Friehs, G., Henderson, J.M., Shenoy, K.V., Donoghue, J.P., Hochberg, L.R., 2015. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. Sci. Transl. Med. 7, 313ra179-313ra179. https://doi.org/10.1126/scitranslmed.aac7328

Simeral, J.D., Kim, S.-P., Black, M.J., Donoghue, J.P., Hochberg, L.R., 2011. 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. https://doi.org/10.1088/1741-2560/8/2/025027

Third, given the paper's emphasis on how the character and word decoding rates surpass existing state of the art, the data may actually have much more information about the nature of neural representation of attempted handwriting that could benefit a broader audience (particularly the neurobiology and neurophysiology communities), but this is not emphasized in the current version of the paper. As such, it is unclear if the work will be of immediate interest to many people from several disciplines.

Thank you for this suggestion. We appreciate the desire to understand how handwriting is neurally represented and what this might mean for the cortical motor system in general. We are currently working on a separate manuscript to accomplish this goal. Since there are already numerous BCI-related results and methods that we must cover in this manuscript, we believe that it is best to retain the current focus on the BCI aspects.

A BCI-centered focus keeps within the tradition of previous "first-of" BCI papers (examples referenced below), which have all achieved wide interest and impact by focusing largely on their BCI achievement. Additionally, we believe that the computational richness of the problem of neurally decoding sequences of handwriting movements, combined with a public release of this unique dataset, should attract broad interest across the machine learning community as well.

Ajiboye, A.B., Willett, F.R., Young, D.R., Memberg, W.D., Murphy, B.A., Miller, J.P., Walter, B.L., Sweet, J.A., Hoyen, H.A., Keith, M.W., Peckham, P.H., Simeral, J.D., Donoghue, J.P., Hochberg, L.R., Kirsch, R.F., 2017. Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration. The Lancet 389, 1821–1830. https://doi.org/10.1016/S0140-6736(17)30601-3

Anumanchipalli, G.K., Chartier, J., Chang, E.F., 2019. Speech synthesis from neural decoding of spoken sentences. Nature 568, 493–498. https://doi.org/10.1038/s41586-019-1119-1

Bouton, C.E., Shaikhouni, A., Annetta, N.V., Bockbrader, M.A., Friedenberg, D.A., Nielson, D.M., Sharma, G., Sederberg, P.B., Glenn, B.C., Mysiw, W.J., Morgan, A.G., Deogaonkar, M., Rezai, A.R., 2016. Restoring cortical control of functional movement in a human with quadriplegia. Nature 533, 247–250. https://doi.org/10.1038/nature17435

Collinger, J.L., Wodlinger, B., Downey, J.E., Wang, W., Tyler-Kabara, E.C., Weber, D.J., McMorland, A.J., Velliste, M., Boninger, M.L., Schwartz, A.B., 2013. High-performance neuroprosthetic control by an individual with tetraplegia. The Lancet 381, 557–564. https://doi.org/10.1016/S0140-6736(12)61816-9

Hochberg, L.R., Bacher, D., Jarosiewicz, B., Masse, N.Y., Simeral, J.D., Vogel, J., Haddadin, S., Liu, J., Cash, S.S., Smagt, P. van der, Donoghue, J.P., 2012. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. Nature 485, 372–375. https://doi.org/10.1038/nature11076

Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Chen, D., Penn, R.D., Donoghue, J.P., 2006. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature 442, 164–171. https://doi.org/10.1038/nature04970

Fourth, direct comparison to behaviors requiring dexterous movements such as typing at speeds of 120 characters per minute for intact subjects is somewhat irrelevant since the ability to modulate brain signals to become a reliable source of control of these assistive devices vary considerably among human subjects who cannot move or speak. For example, it is unclear that the achieved speed/error rates will generalize to other subjects with similar impairment. In other occasions, they draw comparison to speech-decoding BCIs for restoring verbal communication, but this technology is at a very early stage to be compared to the current approach.

Thank you for highlighting this important point. Indeed, subject-to-subject variability is an important issue in BCI research, especially for a single-subject study. In our Discussion section, we now more explicitly mention that this is a limitation of the current work (reproduced below for convenience):

> Finally, it is important to recognize that ~~our~~ the current system is a proof-of-concept that a high-performance handwriting BCI is possible (in a single participant capable of handwriting prior to his injury); it is not yet a complete, clinically viable system. More work is needed to demonstrate high performance in additional people, expand the character set (e.g. capital letters), enable text editing and deletion, and maintain robustness to changes in neural activity without interrupting the user for decoder retraining. More broadly, intracortical microelectrode array technology is still maturing, and requires further demonstrations of longevity, safety, and efficacy before widespread clinical adoption[33,34]. Variability in performance across participants is also a potential concern that may require improvements in intracortical recording technology to increase consistency (in a prior study, T5 achieved the highest performance of 3 participants[7]).

Again, we agree that subject-to-subject variability is an important issue to highlight (as per above), and we also believe that it is helpful to readers to place these BCI typing rates into a broader context by comparing them to able-bodied typing rates. Comparing to able-bodied typing rates can help the reader better appreciate how fast the current BCI typing rates are, and how much of a gap between BCI performance and able-bodied typing remains. We think that BCI research should seek to achieve communication rates that are as close to able-bodied communication rates as possible, as presumably this gives the most benefit to the user (although it may not always be possible to do so).

Regarding speech-decoding BCIs, we thought that it would offer the reader valuable context to briefly review other types of communication BCIs and how they compare with the handwriting BCI. For example, readers might wonder whether there is value in a handwriting BCI if a speech BCI can restore communication at much faster speeds. We think it is therefore appropriate to let the reader know that although speech is faster than handwriting, no speech BCI has yet demonstrated accuracies high enough to restore general-purpose communication. We briefly mention speech BCIs only once, in the following sentence in the Discussion (which we have re-worded in a more positive way):

> Recently, speech-decoding BCIs have shown exciting promise for restoring rapid communication (e.g. [32,17,18]), but their accuracies and vocabulary sizes require further improvement to support~~ies are currently too limited for~~ general-purpose use.

Taken together, the authors should present their findings within the broader context in which the population of potential beneficiaries need to opt for a brain surgery with unknown longevity of the implanted device and a relatively long calibration process to gain additional typing speeds (extra 33 characters/min as I consider the self-paced performance reported here to be the real use case of such communication technology).

Thank you for this important suggestion to address the broader context. We have done our best to place this work into the broader context of intracortical BCI technology. The Discussion now mentions both the current limitations of intracortical BCIs and the reasons to be optimistic about how the technology may continue to evolve. In addition, we have addressed the calibration issue directly, as described above, and

now highlight it more extensively in the Discussion and Results. We believe that our new analyses demonstrate that a long calibration process is likely not necessary.

Regarding brain surgery and array longevity, we believe that a product should be brought to market only after safety and efficacy clinical trial studies systematically demonstrate array safety and longevity. We do not advocate that the general patient population opt for a medical product of unknown longevity/efficacy, and of course FDA approval would be required before this is even possible. As is standard practice with clinical trials, only participants who clearly understand that there is no benefit assured if they elect to be a part of an early clinical trial (e.g., BrainGate) should consider providing informed consent and participating in the clinical trial. To help better assess array longevity, we are currently preparing a manuscript that summarizes longevity and efficacy data systematically across all 14 participants in the BrainGate pilot clinical trials. Similarly, we envision that neurotechnology companies (e.g. Synchron, Neuralink, Paradromics) will (and must) conduct systematic trials to evaluate the safety and longevity of any new electrode device before a product is released. We have thus chosen to structure the Discussion with an eye towards the fact that more safety and longevity data will be collected in the future, as opposed to weighing the current lack of such data as a disadvantage for the handwriting BCI. We view our work as providing additional motivation to collect such data and for research groups to pursue this line of research (and for companies to pursue such a product).

Finally, we would like to briefly clarify that, to our knowledge, the current BCI typing record for free-typing (as opposed to copy-typing) is 24 characters per minute. This record was set by an intracortical point-and-click BCI (Pandarinath et al. *eLife* 2017). Thus, our current free-typing rate of 73 characters per minute is an extra 49 characters per minute (three-fold increase).

C. Data & methodology:

General comments:

The presentation is clear, logical and readable to general audience. The reporting of data and methodology is sufficiently detailed to enable reproducing the results. They state that they will share the data and code to enable reproducibility.

Thank you.

Major Comments:

The authors state that they ‘linearly decoded pen tip velocity from neural activity’. Arguably, this variable varies considerably among different people depending on their handwriting style, accuracy, appearance, readability, etc. Did the authors have a sample handwriting from the subject before injury so they can be compared to the ones they decoded? If so, could they analyze such data to infer the pen tip speed profiles the subject likely used to better understand if the observed neural activity correlated with the character shapes? it would be more helpful if the work attempts to provide some understanding of the extent to which the dynamics of the ensemble neural activity do actually reflect this critical behavioral parameter.

Thank you for this interesting idea. Unfortunately, we did not have any handwriting samples readily available to us, as T5’s injury occurred 9 years prior to this study (otherwise we would have proceeded as you describe). Instead, we describe below what we did do, but it in greater detail so that it is clearer (and we have also added detail to the manuscript to make it clearer as well).

To understand T5’s writing style, we interviewed T5 about how exactly he wrote each letter. Then, we used a computer mouse to trace the trajectory of each letter in the same way that T5 reported doing so (while recording the X & Y velocity of the mouse pointer). These trajectory templates, *which are time series of velocity vectors* (not spatial drawings), were then used to train a linear decoder to decode the pen tip velocity. Although these trajectories cannot be expected to match T5’s trajectories in a precise way, they should nevertheless capture the general features of each letter trajectory. Figure 1D,

reproduced below for convenience, shows the output of linear decoders trained to decode pen tip velocity using these templates:

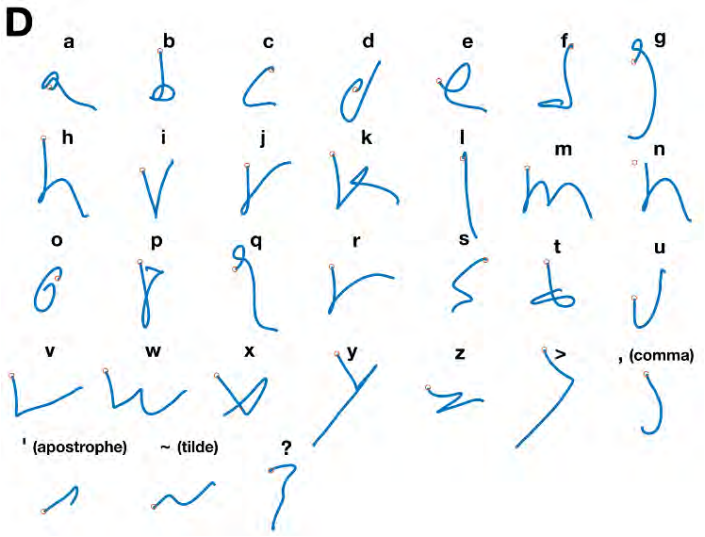


Fig. 1D. Decoded pen trajectories are shown for all 31 tested characters: 26 lower-case letters, commas, apostrophes, question marks, tildes (~) and greater-than signs (>). Intended 2D pen tip velocity was linearly decoded from the neural activity using cross-validation (each character was held out). The decoded velocity was then averaged across trials and integrated to compute the pen trajectory (orange circles denote the start of the trajectory).

Importantly, these letter reconstructions were *held-out* reconstructions. In other words, each letter shape was reconstructed using a decoder that was trained only on *other* letters. The output of each velocity decoder was then cumulatively integrated to compute a pen tip position trajectory, which was drawn as the character reconstruction in Fig.1D. The fact that recognizable letter shapes were decoded demonstrates that there was a consistent neural encoding of pen tip velocity. Otherwise, the decoders might have been able to overfit to the training data but would not have been able to reconstruct pen tip velocity correctly for held-out characters, resulting in unrecognizable shapes. It is worth noting that the reconstructed pen trajectories are well correlated with the letter templates (r = 0.74 across all held-out reconstructions).

In the original manuscript, this important detail about decoder training was mentioned only in the figure legend and Methods. We now clarify in the Results text that reconstructions were only made with decoders *not* trained on that character:

> Readily recognizable letter shapes confirm that pen tip velocity is robustly encoded (each character's reconstruction was made using a decoder trained only on *other* characters).

We also amended the Methods section to add more detail:

> To train the decoder, we used hand-made templates that describe each character's pen trajectory. The character templates were made by drawing each character with a computer mouse in the same way as T5 described writing the character. As each character was drawn, the X and Y velocity trajectories of the mouse pointer were recorded.~~;~~ T~~t~~hese templates (which are time series of velocity vectors) then defined the target velocity vector for the decoder on each time step of each trial. We used ordinary least squares regression to train the decoder to minimize the error between the template velocities and the decoded velocities (see Supplemental Methods for more details). The reconstructed pen tip velocities that were decoded in Fig. 1D were well correlated with the mouse templates (r = 0.74 across all characters).

Next, to understand how large the neural encoding of pen tip velocity was compared to other elements of the neural activity, we used a linear encoding model to fit neural activity as a function of the reconstructed pen tip velocity. This quantifies how much of the neural activity is captured by the neural dimensions that encode pen tip velocity. We found that 30% of the variance was accounted for by pen tip velocity. This is a sizeable portion but still leaves much of the neural activity unaccounted for, which is consistent with recent studies that have highlighted non-kinematic aspects of motor cortical activity [1-2]. We now report this in the Results:

The neural dimensions that represented pen tip velocity accounted for 30% of the total neural variance.

[1] Kaufman, Matthew T., Jeffrey S. Seely, David Sussillo, Stephen I. Ryu, Krishna V. Shenoy, and Mark M. Churchland. "The Largest Response Component in the Motor Cortex Reflects Movement Timing but Not Movement Type." *ENeuro* 3, no. 4 (July 1, 2016): ENEURO.0085-16.2016. https://doi.org/10.1523/ENEURO.0085-16.2016.

[2] Churchland, Mark M., John P. Cunningham, Matthew T. Kaufman, Justin D. Foster, Paul Nuyujukian, Stephen I. Ryu, and Krishna V. Shenoy. "Neural Population Dynamics during Reaching." *Nature* 487, no. 7405 (July 5, 2012): 51–56. https://doi.org/10.1038/nature11129.

Also, the authors should demonstrate the extent to which character encoding might have changed as a function of trials/sentences/sessions, particularly during times when the subject was observing the prompted text, the decoded text, and when the subject was asked to write from memory. This characterization is also needed to provide credence for the claim made in the conclusion that this is a BCI without visual feedback.

Thank you for this suggestion. Due to this suggestion and others below, we have now removed the claim that our BCI can function without visual feedback. While we did collect some data with his eyes closed, it is not a major point and we believe that it is better to remove this to help the manuscript stay focused.

We think that analyzing differences in neural tuning across contexts is a valuable direction for future work, but one that lies outside the scope of the current study, as it does not directly bear on the central claims of this manuscript. Analyzing how characters are neurally encoded during sentence writing is also a difficult task, as it requires accurate segmentation of unlabeled data. Although we have solved this problem well enough for BCI decoding, it is unclear whether small errors in data segmentation could cause artifactual differences in neural encoding to appear.

Given the data we have shown, we would propose that the neural encoding must be at least broadly similar across contexts, since decoders trained on open-loop data (where T5 is copying sentence prompts but no BCI is active) can transfer accurately to the closed-loop context (where T5 is using the BCI and observing the decoded text appear on the screen). Nevertheless, we do agree that differences in neural coding across contexts may have been the cause of some decoding errors, and that it would be worthwhile and interesting to pursue this possibility in future work focused on addressing this question.

It is unclear if the authors have characterized the performance long enough (beyond the stated 10 sessions) to report how nonstationarity in the neural signals can potentially deteriorate the performance reported. In fact, with the exception of the first couple of sessions that were spaced almost a month apart, the remaining 9 sessions took place almost 6 months afterwards and were closely spaced, happening within the span of 7-8 weeks. From the extensive calibration protocol described, there seems to be substantial variability in these signals.

Thank you again for this helpful suggestion. As we laid out above, we believe the new analyses on nonstationarity and decoder calibration provide useful insight into these questions. We think that the 10 sessions we reported, which comprise the entirety of our data, are sufficient for preliminary estimates of nonstationarity and the amount of calibration data required for decoder training.

More specifically, closely-spaced sessions are the most relevant to this question, as we imagine that this kind of BCI will be used at least once every few days or possibly weeks. Given the size of neural

nonstationarities that accrue on intracortical electrode arrays over long time spans (e.g. several months) [1], we don't expect decoders to be able to retain high performance after months of time have passed with no recalibration (at least with the current state of electrode array technology). Our new supplemental figure confirms this. It shows that for sessions 6 months apart, changes in neural activity are substantial (SFig. 4A-B). However, these new figures also demonstrate that for more closely-spaced sessions, good performance can be achieved with no recalibration at all, or unsupervised recalibration that need not consume any user time (as it can run in parallel during normal use). We envision a usage scenario involving light recalibration (or unsupervised recalibration) during periods of regular use, combined with larger calibration datasets when users return from months of inactivity.

We understand, however, that there is a desire (and a need) to characterize nonstationarities systematically across many more subjects and larger datasets. We believe that the best way to do this is with a comprehensive study of many participants spanning many years. We are currently in the process of quantifying how neural signals recorded on intracortical electrode arrays change over time using data from all 14 BrainGate participants over a time span of 15 years, which we plan to report in a future publication. In this way we believe that we will be able to most meaningfully, and rigorously, contribute new insight on this important question.

[1] Downey, John E., Nathaniel Schwed, Steven M. Chase, Andrew B. Schwartz, and Jennifer L. Collinger. "Intracortical Recording Stability in Human Brain–Computer Interface Users." *Journal of Neural Engineering* 15, no. 4 (May 2018): 046016. https://doi.org/10.1088/1741-2552/aab7a0.

Specific comments:

Line 93: Why did the subject write 'periods as '~' and spaces as '>'?

Thank you, we should have clarified. We instructed T5 to write periods with a '~' symbol because we thought that '~' would be easier to detect than just a single dot. Similarly, we wanted to associate spaces with a symbol instead of just the absence of writing. The '~' and '>' symbols were chosen with an eye towards being easy to write and classify, but were not the result of a systematic study of which symbols would be the best. We added the following sentence of explanation to the Results section:

> The '~' and '>' symbols were chosen to make periods and spaces easier to detect.

Line 100: Clarify if the statement 'After each new day of decoder evaluation,' refers to offline or online decoding.

Thank you. Decoders were calibrated each day *before* they were evaluated online, using data collected from that day combined with all prior days.

We re-worded that section to now state the following:

> Prior to the first day of real-time use described here, we collected a total of 242 sentences across 3 days that were combined to train the RNN (sentences were selected from the British National Corpus). On each day of real-time use, additional training data was collected to retrain the RNN prior to real-time evaluation, yielding a combined total of 572 training sentences by the last day (comprising 7.3 hours and 30.4k characters). ~~After each new day of decoder evaluation, that day's data was cumulatively added to the training dataset for the next day (yielding a total of 572 sentences by the last day).~~

Line 112: How did the authors know the exact timing of completion of each letter by the subject in real time to be able to display it after it was completed? It is stated that visual feedback about the decoder output was 'estimated to be between 0.4-0.7'. The supplementary material explains how they arrived at these estimates, but this inherently assumes that the character was 'completed' when the start of a new one was detected. One can argue that natural handwriting of a word does not entail separating in time the representation of characters — they are all 'connected'.

Thank you, this is indeed an important and somewhat complex aspect that we should have explained more clearly. During real-time use, a simple thresholding scheme was used to decide when to decode and display each letter to the screen. Specifically, the RNN's "new character" output (see Fig 2A, reproduced below) was thresholded (threshold = 0.3). Whenever it crossed the threshold at time *t*, the most likely character at time *t*+0.3s was emitted. The most likely character was determined by examining the RNN's 'character' output.

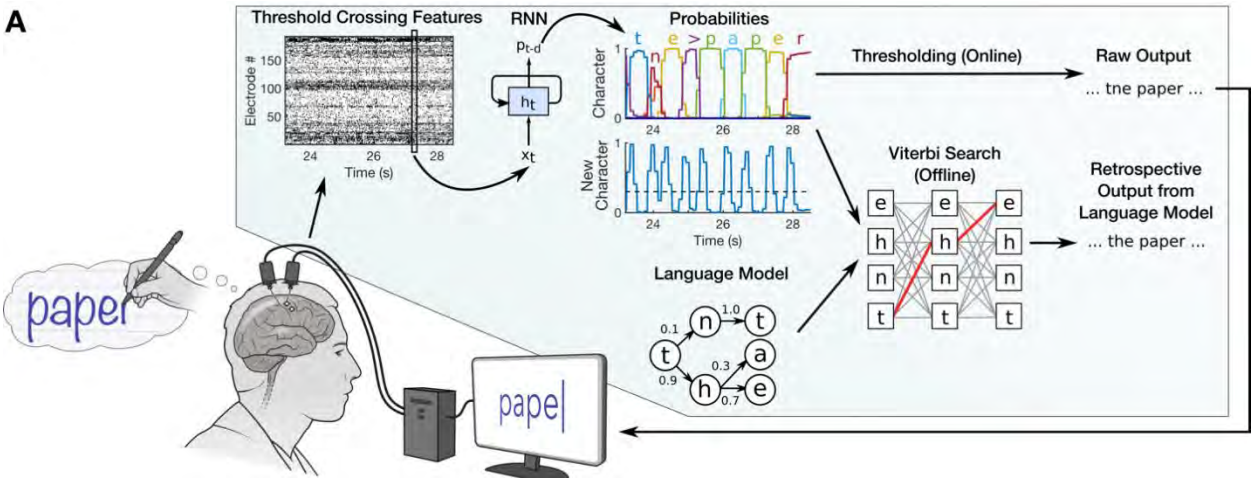


Figure 2A. Diagram of our decoding algorithm. First, the neural activity (multiunit threshold crossings) is temporally binned (20 ms bins) and smoothed on each electrode. Then, a recurrent neural network (RNN) converts this neural population time series (x_t) into a probability time series (p_{t-d}) describing the likelihood of each character and the probability of any new character beginning. The RNN has a one second output delay (d) so that it has time to observe the full character before deciding its identity. Finally, the character probabilities were thresholded to produce "Raw Output" for real-time use (when the "new character" probability crossed a threshold at time t, the most likely character at time t+0.3s was emitted and shown on the screen). In an offline retrospective analysis, the character probabilities were combined with a large-vocabulary language model to decode the most likely text that the participant wrote (we used a custom 50,000-word bigram model).

Given the absence of ground truth data about T5's attempted pen movements, we can only offer a "best guess" of the visual latency. In some sense, answering this question with certainty would require a complete solution to the original decoding problem posed here: segmentation and classification of characters from an unlabeled data stream. Given that our RNN decoder is not perfect, the RNN outputs can only offer a rough estimate of the latency.

Regarding the possibility of 'connected' characters, we do appreciate that there is some ambiguity and arbitrariness in defining exactly when a character ends and another begins, and that there is likely some 'transition time' which occurs between any two characters. Mitigating this issue somewhat is the fact that T5 reported writing each character in a print (not cursive) font, with each letter printed directly on top of the previous one as if writing on a PalmPilot. We added the following clarification to the Results section:

> T5 attempted to write each character in print (not cursive), with each character printed on top of the previous one.

Finally, it is unclear to us how exactly we could modify the estimated latency to account for the potential time spent transitioning between letters (since this transition time is unknown); as such, and after much discussion, we decided it would be best to keep the estimate as-is, with the understanding that it is only an estimate.

One can also argue that their approach (delaying the decoder output by 1 sec and adding the filter kernel widths to the total interval) prevents visual feedback about the state of neural activity until a complete character is encoded by the subject, but the reality is that the subject can 'covertly' infer information from the structure of the word being typed (self-generated case) and visual feedback from the screen (on-prompt case).

Thank you for pointing this out, we see now that visual feedback of the prompt and/or previously decoded letters could be used by T5. Due to this suggestion and others, we have removed any claim that our BCI can operate without visual feedback.

Line 115: How did neural activity look like when an error was made? and when the subject was provided visual feedback about the language model's autocorrection of that error? Did the subject stop modulating, eventually relying on the model to autocorrect, or did he continue to modulate neural activity to correct the typo? Was the decoder 'disengaged' in those instances? did the neural activity occupy different regions of the state space relative to the intended character or the corrected character?

Thank you for these interesting questions and suggestions. First, we want to clarify that the language model was only applied *offline* in a retrospective analysis and was never used online (i.e., T5 never saw the results of the language model).

Since there was no 'backspace' implemented, T5 was simply instructed to ignore errors and continue uninterrupted. T5 reported spending most of his time looking at the prompt during the copy-typing task, instead of watching the decoded letters appear on the screen and scanning for errors. We confirmed this using an eye tracker. Analyzing eye position data, we found that during copy-typing T5 spent 93% of the time looking at the prompt. T5 looked at the decoded text mostly at the end of each trial after all characters had been typed (but before he triggered the beginning of the next trial). During this "end-of-trial" period, T5 spent 82% of the time looking at the decoded text.

We added the following details to the Results section:

> Since there was no 'backspace' function implemented, T5 was instructed to continue writing if any decoding errors occurred. T5 reported spending most of his time looking at the prompt instead of watching the decoded letters appear on the screen (eye tracking data confirmed that T5 spent 93% of the time looking at the prompt; Tobii 4C eye tracker).

As suggested by the reviewer, it is interesting to ask how the perception of errors affect the neural activity. Recent reports from our group suggest that errors cause a distinct neural signature in motor cortex that can even be detected with a BCI and used to 'undo' errors [1-2]. We think this is an interesting line of future research for handwriting decoders.

[1] Even-Chen, Nir, Sergey D. Stavisky, Jonathan C. Kao, Stephen I. Ryu, and Krishna V. Shenoy. "Augmenting Intracortical Brain-Machine Interface with Neurally Driven Error Detectors." *Journal of Neural Engineering* 14, no. 6 (November 2017): 066007. https://doi.org/10.1088/1741-2552/aa8dc1.

[2] Even-Chen, N., S. D. Stavisky, C. Pandarinath, P. Nuyujukian, C. H. Blabe, L. R. Hochberg, J. M. Henderson, and K. V. Shenoy. "Feasibility of Automatic Error Detect-and-Undo System in Human Intracortical Brain–Computer Interfaces." *IEEE Transactions on Biomedical Engineering* 65, no. 8 (August 2018): 1771–84. https://doi.org/10.1109/TBME.2017.2776204.

Line 118: It is stated that the raw decoder output plateaued at 90 characters per minute with a 5.4% character error rate. But the comparison drawn in the sentence that followed argues that the 'word error rate' decreased to 3.4% average across all days. The authors should provide the reduction in 'character error rate' not 'word error rate' with the use of the language model to make this comparison objective. Arguably, many words share the same characters and understanding of words depends on the sentence context.

Thank you, we now mention both character error rate and word error rate in the Results text:

> Importantly, typing rates were high, plateauing at 90 characters per minute with a 5.4% character error rate (Fig. 2C, average of red circles). When a language model was used to autocorrect errors, error rates decreased considerably (Fig. 2C, open squares below filled circles; Table 1). The character error rate fell to 0.89% and the word error rate fell to 3.4% averaged across all days, which is comparable to state-of-the-art speech recognition systems (e.g. word error rates of 4-5% [15,16]) …

Line 120: it is stated that 'a new RNN was trained using all available sentences to process an entire sentence'. This means that offline decoding of an entire sentence achieved the stated 0.17% character error rate. As stated this decoder has not been used by the subject in real time to see if this newly trained decoder will be able to display an entire sentence at the end of a neural activity modulation epoch by the subject in the absence of the delayed character-by-character feedback as in the online case. As such, what is the significance of this result?

Thank you, we should have been clearer. In this analysis, we trained a bidirectional, acausal RNN to use *all* of the neural data in a sentence in order to decode that sentence (as opposed to using, for each time point *t*, only data that occurred prior to *t* (i.e., causal)). We see the significance of this result as two-fold: (1) providing a point of comparison to other work in the BCI and machine learning fields that process neural activity, handwriting or speech in an acausal manner, and (2) demonstrating a high ceiling for accurate performance, meaning that the trial-to-trial neural variability is not too great to prevent very high decoder performance.

As we see it, this result is mainly to provide more context and insight into the data, not necessarily to suggest that such a decoder be used in real-time as part of the BCI (which, as the reviewer points out, would not give the user character-by-character feedback). Nevertheless, it is possible to combine the causal decoder with the bidirectional decoder. One could use the causal decoder to give character-by-character feedback, and then run the bidirectional decoder at the end of each sentence to further clean up any decoding errors.

We added the following additional explanation to the Results section:

> Finally, to probe the limits of possible decoding performance, we retrospectively trained a new RNN using all available sentences to process the entire sentence in a non-causal way (comparable to other BCI studies [17,18]). In this regime, accuracy was extremely high (0.17% character error rate averaged across all sentences), indicating a high potential ceiling of performance. Although such an acausal decoder would not be able to provide letter-by-letter feedback to the user, it could be used to correct errors after the user finishes typing a sentence.

Table 1: Can the authors explain why the word error rate is so high (25.1%) in the raw online output case despite a character error rate of 5.9%?

Under the standard definition of word error rate, a word is incorrect if *any* character in that word is incorrect. On average, English words have five characters in them. Thus, with a character error rate of 5.9%, if we assume that each character independently has a 94.1% chance of being accurate, we might expect a word error rate of $1-(0.941)^5 = 26.2\%$. We added the following explanation to the table caption:

> Word error rates are high for "online output" because a word is considered incorrect if *any* character in that word is wrong.

Supplementary material:

Line 427: it is stated that "some micromotions of the right hand were visible during attempted handwriting (see 10 for neurologic exam results and SVideo 4 for hand micromotions). Have the authors quantified the extent of variance in the neural data that could be explained by this potential confound?

Thank you for raising this interesting question, which we have considered by did not clarify in the original manuscript. In our view, the potential leakage of motor commands into small amounts of muscle activity is not a confound here. First, we have added additional text to clarify the extent of T5's injury and paralysis, which is severe.

The description in the Results now reads:

> T5 has a high-level spinal cord injury (C4 AIS C) and was paralyzed from the neck down; his hand movements were entirely non-functional and limited to twitching and micromotion.

In the Methods section, we have added neurological exam data:

> Below the injury, T5 retained some very limited voluntary motion of the arms and legs that was largely restricted to the left elbow; however, some micromotions of the right hand were visible during attempted handwriting (see [12] for full neurologic exam results and SVideo 4 for hand micromotions). T5's neurologic exam findings were as follows for muscle groups controlling the motion of his right hand: Wrist Flexion=0, Wrist Extension=2, Finger Flexion=0, Finger Extension=2 (MRC Scale: 0=Nothing, 1=Muscle Twitch but no Joint Movement, 2=Some Joint Movement, 3=Overcomes Gravity, 4=Overcomes Some Resistance, 5=Overcomes Full Resistance).

Thus, we believe that T5 is a good / reasonable model of someone who could benefit from a communication BCI – i.e., someone who might be able to generate some hand micromotions but retains essentially no hand function. In our experience, severe paralysis is rarely fully complete. This is supported by a recent study of potential BCI users [1] that found that "incomplete" locked-in syndrome, which still prevented normal communication due to severe paralysis, was significantly more common than complete locked-in syndrome.

[1] Pels, Elmar G.M., Erik J. Aarnoutse, Nick F. Ramsey, and Mariska J. Vansteensel. "Estimated Prevalence of the Target Population for Brain-Computer Interface Neurotechnology in the Netherlands." *Neurorehabilitation and Neural Repair* 31, no. 7 (July 2017): 677–85. https://doi.org/10.1177/1545968317714577.

Second, we believe that the neural activity is generated primarily by the *intention* to move, and not overt motion itself. This is supported by a recent study from our group which included data from participant T5; in that work, we found that body parts which T5 still had control over (e.g. head, shoulder) did not have a stronger representation than body parts which were fully or almost fully paralyzed [2]. This view is also supported by a previous study on point-and-click BCIs from our group that included a participant (T6) who still retained hand function (and used thumb/index finger motor imagery to control the cursor). We found that we could achieve high performance whether or not the participant made overt finger motions, suggesting that the neural activity was primarily driven by motor intent and not, for example, sensory feedback generated by overt motion [3].

[2] Willett, Francis R., Darrel R. Deo, Donald T. Avansino, Paymon Rezaii, Leigh R. Hochberg, Jaimie M. Henderson, and Krishna V. Shenoy. "Hand Knob Area of Premotor Cortex Represents the Whole Body in a Compositional Way." *Cell*, March 26, 2020. https://doi.org/10.1016/j.cell.2020.02.043.

[3] Pandarinath, Chethan, Paul Nuyujukian, Christine H. Blabe, Brittany L. Sorice, Jad Saab, Francis R. Willett, Leigh R. Hochberg, Krishna V. Shenoy, and Jaimie M. Henderson. "High Performance Communication by People with Paralysis Using an Intracortical Brain-Computer Interface." *ELife* 6 (February 21, 2017): e18554. https://doi.org/10.7554/eLife.18554.

We added the following explanation to the *Methods* section where T5's injury is described in detail:

> In a recent study from our group which included data from participant T5, we found that body parts which T5 still had control over (e.g. head, shoulder) did *not* have a stronger representation than body parts which were fully or almost fully paralyzed[1]; thus, T5's limited hand motion likely did not have a large effect on the neural activity, which seems to be generated primarily by the *intention* to move and not overt motion itself.

Line 491: It would be informative for the authors to comment on how did the neural activity differ between repetitions of each character individually and when they are within a word or a sentence.

Thank you for this suggestion. While very interesting, as we articulated above, we think that examining the neural encoding of characters within words and sentences is not trivial, since it may introduce artifacts due to imperfect segmentation of words and sentences. Also, as it does not directly bear on the claims made in this manuscript, we prefer to leave this analysis for future work. Nevertheless, we do appreciate that examining how the context in which characters are written affects neural encoding is a useful and important direction that may yield decoder performance improvements and basic neuroscience insight.

D. Appropriate use of statistics and treatment of uncertainties:
Figures are well illustrated. Probability values and error bars are explained. There were no statistical significance tests performed.

Thank you.

Line 178: Authors should provide more explanation for "the participation ratio (PR), which quantifies approximately how many spatial or temporal axes are required to explain 80% of the variance in the neural activity patterns" in this section. Readers have to refer to the supplementary methods section to understand this metric.

Thank you for this suggestion. We do appreciate the desire to have every metric clearly explained as it is introduced in the Results. However, we think that referring to the Methods section, at least some of the time, is unavoidable in a Results-first format that is highly space-constrained like *Nature*. In our mind, to understand the result readers only need to know that this is a continuous quantification of dimensionality. However, to understand how the metric is computed seems to require an equation and a paragraph-sized description, which doesn't fit in the Results. Note that the metric is explained in the *Methods* section, not the *Supplementary Methods* (which are in an entirely separate document that contains much more detailed protocols).

We now offer the following additional clarification and refer the reader to the Methods section:

> Spatial and temporal dimensionality were quantified using the participation ratio (PR), which ~~quantifies~~ is a continuous quantification of approximately how many spatial or temporal axes are required to explain 80% of the variance in the neural activity patterns[21] (see Methods for details).

Line 192 Figure 3: The authors find that increased temporal complexity in neural state space trajectories could make movements easier to decode compared to trajectories that do not have such complexity, or have only spatial complexity. They then present a toy example in Figure 3 to make this point. I would partly disagree with their assessment and argument for the following reasons:

i) In the toy example in (Figure 3F) they increased variations of neural trajectories over time to illustrate that this increases separability (measured by nearest neighbor distance) compared to the case where the neurons' activity is constrained to a single spatial dimension, the unity diagonal). But the example lacks inclusion of noise, the temporal characteristics of which can easily 'fool' the classifier, making it think there is more temporal complexity in the trajectories than really is.

ii) The nearest neighbor distance and consequently classifier performance should be characterized when noise is present in this toy example, with a parameter that controls the amount of temporal complexity in noisy neural trajectories. Directions of fluctuations around these trajectories are likely to influence the conclusion made, both in the straight line as well as the handwritten characters cases.

Thank you for these important suggestions. Below we expand on our approach and rationale, and while in principle we are very open to adding this entire treatment to the manuscript's supplemental materials, we are facing space limitations such that we would need to seek guidance on how to be able to do this and if

it is possible at all. Thus, we thought that we would provide this explanation here and potentially go from there if there is still a need to do so.

First, we would like to clarify that this toy example does include noise. The three panels (F, G, H) from Fig. 3 (now Fig. 4) are reproduced below for convenience. Panel H shows how classification accuracy varies as a function of the amount of neural noise present (simulated as white noise). When there is no noise present, it is trivially easy to classify between the four conditions because there is no chance that one could be confused for another. Panel H shows that in the no-noise case (σ=0), there is no difference between the classification performance of "simple" trajectories (shown in F) and "complex" trajectories (shown in G) because classification performance is 100% for both. However, as the amount of noise increases, complex trajectories become easier to classify because their nearest neighbor distances are larger (and thus nearby trajectories are less likely to be confused with each other).

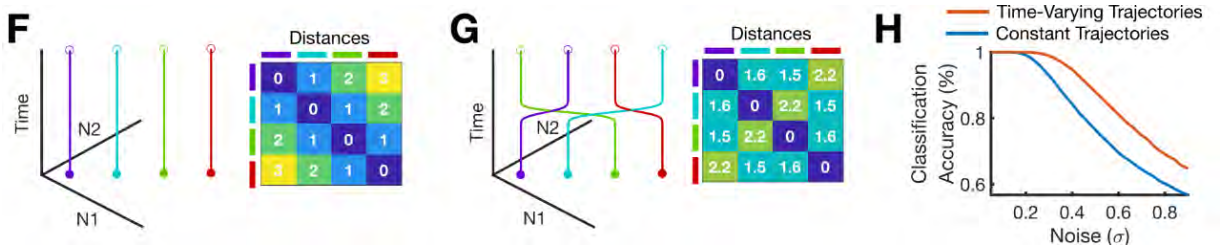


Fig. 3F-H. (Now Fig. 4).

Note that the temporal complexity (dimensionality) of the noise is much higher than that of the trajectories themselves. By definition, white noise occupies all possible temporal dimensions. In this toy example, we discretized the trajectories into 100 time steps; thus, the temporal dimensionality of the white noise was 100. The temporal dimensionality of the underlying neural trajectories themselves was much lower (1 for the simple trajectories, 2 for the complex trajectories). Why is the temporal dimensionality of the noise so high? Because white noise is independent for each time step, it requires one dimension for each time step in order to fully describe it. The underlying neural trajectories, on the other hand, are much smoother across time.

We added the following clarifications to the Methods section:

> … Thus, we performed the simulated classification ~~on~~ using the true neural patterns themselves (but still in the presence of observation noise). The simulated trajectories were discretized into 100 time steps and white noise was added to each time step independently.

Why, then, is the classifier not confused by the temporal complexity in the noise? To understand this, it may help to define some terms. Let f_a be a vector that describes the underlying neural trajectory for movement *a* (i.e., the mean neural firing rates across time for movement *a*). Each entry in the vector f_a is the mean firing rate for a given time step (to describe multiple neurons, the activity profile of each neuron can be stacked one on top of the other in the vector). Let ε be a neural noise vector for an example trial of movement *a*. The observed neural activity on that trial is then f_a+ ε. A classifier confusion will only happen for this trial if f_a+ ε looks more like the mean firing rates for a different movement (e.g. f_b) than it looks like f_a. We can formalize this notion of "looking like" with Euclidean distance; in other words, a confusion will occur if:

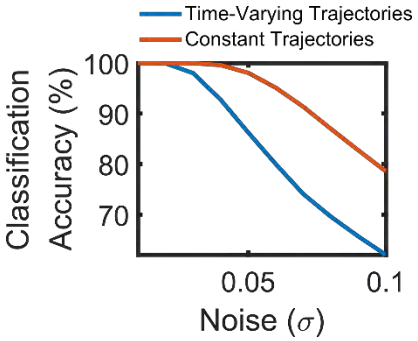
$$\|(f_a + \varepsilon) - f_b\| < \|(f_a + \varepsilon) - f_a\|$$

These confusions can be reduced if f_a and f_b look more different from each other, thereby reducing the chance that ε will corrupt f_a into looking like f_b. In other words, classification performance is improved when nearest neighbor distances are increased. Temporal variety is just one way to increase this distance (spatial variety is another). In our view, the temporal complexity of the noise ε is thus not

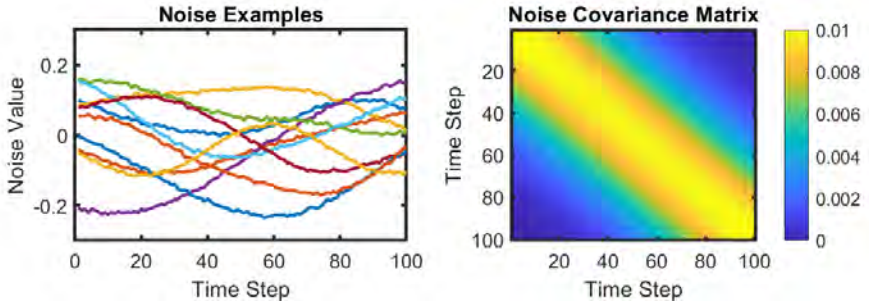
necessarily important here (although its size is – the larger ε is, the greater the chance that it can cause f_a to look like f_b).

One noise property that can end up making a big difference for performance is the "shape" of the noise cloud, or in other words the directions along which ε is particularly concentrated (as suggested by the reviewer). White noise extends equally in all directions, but the most relevant directions for classification are those directions that connect nearby classes (here, this direction would be $f_b - f_a$). This is because, for f_a to be corrupted into looking like f_b, ε must be similar to $f_b - f_a$ (since in this case $f_a + \varepsilon = f_a + f_b - f_a = f_b$). If anything, we think this is actually another reason to prefer movements with higher temporal dimensionality. Larger temporal dimensionality will cause the directions between nearby classes to be more diverse, and less likely to align with directions that contain large amounts of noise. In our experience, large-noise directions typically describe correlated increases and decreases in firing rates across time. Thus, having movements which are more complex in time will make them more robust to correlated noise fluctuations.

To confirm this, we simulated classification performance using "colored" noise with concentrated power in lower frequencies (i.e. correlated noise). The results obtained were the same as in panel H, except with an even greater difference between the time-varying trajectories and constant trajectories:



The plot below shows examples of the noise vectors (to show how they are correlated in time) and the covariance matrix used to generate this noise (by drawing random samples from a multivariate normal distribution). The diagonal band causes nearby time steps to have correlated noise.



Line 244: Authors state that "One unique advantage of our handwriting BCI is that, in theory, it does not require vision (since no feedback of the imagined pen trajectory is given to the participant, and letters appear only after they are completed)." I would argue against that, partially because this claim is contingent on: 1) exact knowledge of the length of time interval where each decoded character is fully known and, 2) the instructed text was always present on the screen in the on-prompt case. To my understanding this was estimated (see my comment on Line 112 above) based on approximations made by the delayed decoder training and time warping algorithm (1.4 sec delay), which was used offline to build spatiotemporal neural "templates" of the characters.

Thank you, as stated above we have removed this claim about visual feedback.

Line 534: Please clarify what is a 'single movement condition'. Is it a character, a word or a sentence? From line 801 it seems it corresponds to character but the earlier sentence needs clarification.

Thank you for pointing this out. Indeed, we had meant to refer to a character. We have rephrased the sentence as follows:

> Next, we used time-warped PCA (https://github.com/ganguli-lab/twpca)[8,9] to find continuous, regularized time-warping functions that align ~~the~~ all trials ~~within a single movement condition~~belonging to the same character together.

Line 553: Authors used character templates drawn by a computer mouse in the same way as T5 described writing the character. This description provides a shape of the character but it is unclear how this information was translated into pen velocity to train the decoder.

Thank you, we should have been clearer about this. As each character was drawn, we recorded the X and Y velocity of the mouse pointer. We have clarified this by adding the following sentence:

> As each character was drawn, the X and Y velocity trajectories of the mouse pointer were recorded.

Line 577: "the criteria for excluding data points from display in Figure 1E is not clear. It is stated that these data labeled as "outliers in each class" were excluded "To make the t-SNE plot clearer". While it is stated that this resulted in removing 3% of data points, the explanation that these "were likely caused by lapsed attention by T5" is not convincing. How did the authors ascertain that this was the case?

Thank you for raising this interesting and important point. T5 reported that he would occasionally fail to complete a trial due to a lapse in attention; however, it is true that there is no easy way to know whether any particular outlier was due to a lapse in attention or some other cause. We have therefore remade the plot with *all* trials included; the result is very similar, save for some outliers that may be distracting but don't change the core result. We reproduce the new Fig. 1E below, and this now appears in the main paper:

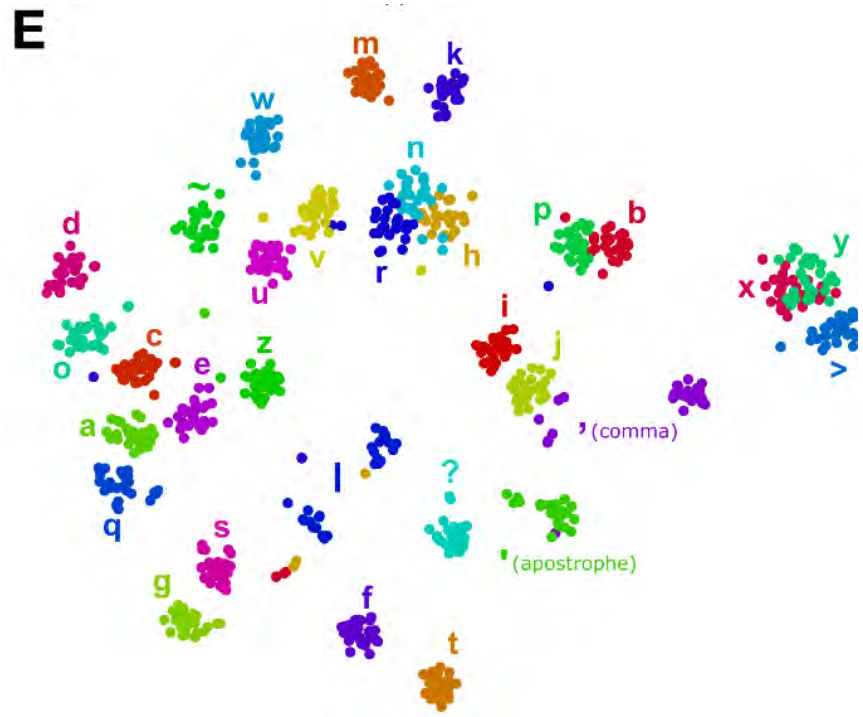


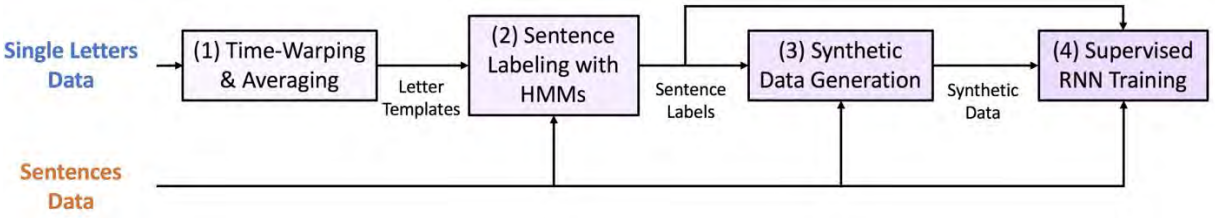
Fig. 1E

Supp Fig 2 and lines 642-667: The authors use a technique from automatic speech recognition literature called forced alignment labeling with HMMs in which they augmented the data via synthetic sentence generation to cope with the limited data size. This section needs improvement regarding how the method works. For example, creating snippets to make synthetic sentences assumes the neural data correspouding to each snippet is independent of the others. How it is then integrated into a new synthetic sentence that is then labeled by the HMM? How 'one-hot representation' is defined based on the heatmaps generated in SF-2D?

Thank you for these questions. Note that there is a detailed explanation of each step in the "Supplemental Methods" document, which is a separate document from the Methods section and contains much more detail about each step. We now make this clearer in the "RNN Training" section in the Methods:

> See SFig. 2B for a diagram of the RNN training flow and Supplemental Methods for a detailed protocol (the Supplemental Methods are contained in a separate online document).

The four steps of RNN training are illustrated in SFig 2B (reproduced below). In step (2), the training data is segmented & labeled with HMMs using a "forced alignment" technique. This step determines, for each time step of data, what character was being written during that time step. In step (3), synthetic sentences are created to 'augment' the data (i.e., these synthetic sentences are added to the original data as additional examples for the RNN to train with). Synthetic sentences are created 'cutting out' the characters from the training data (with the help of the labels from step 2) and placing them into a library of character snippets. These snippets are then re-arranged randomly into new sentences.



SFig. 2B.

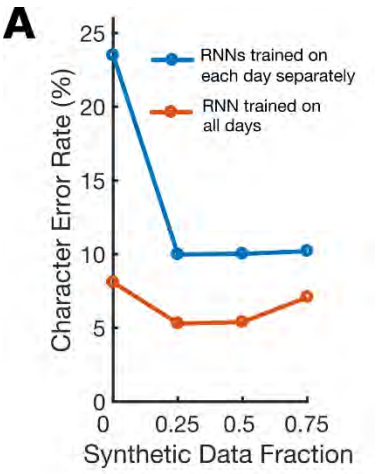
The synthetic sentence creation step does indeed assume that the characters are independent from each other, with one exception: it attempts to choose character examples such that each adjacent pair of characters has matching transition characteristics. This is explained in the Supplemental Methods as follows (in the "Synthesizing the Neural Activity" section):

> For each character, a snippet was chosen from the library at random in a way that attempted to respect pen transition movements between letters. For example, when transitioning from 'e' to 't', the pen must traverse upwards before beginning the downstroke for 't'. However, when transitioning from 'd' to 't', no such pen re-positioning is needed (when written in the way shown in Figure 1). To do this, we discretized the starting heights for each character to the following values: 0, 0.25, 0.5, 1. The assignment of each letter to each category is depicted in the table below.

Start Height	0	0.25	0.5	1.0
Character	comma	a, o, e, g, q	c, d, m, j, i, n, p, r, s, u, v, w, x, y, z, space (>), period (~)	b, t, f, h, k, l, apostrophe, question mark

> When choosing a snippet from the library, we selected at random from all snippets whose next character in the training data began at the same height as the next character in the synthetic sentence. When this wasn't possible, we selected uniformly at random from all snippets.

While these assumptions are simplistic, the main point is that the synthetic data are good enough to significantly improve decoder performance. Supplemental Figure 3A, reproduced below, shows results from an offline analysis that assesses how adding synthetic data reduces the character error rate.



SFig. 3A.

For RNNs trained on a single day, adding synthetic data reduced the character error rate percentage by 12.9 (95% CI = [11.9, 14.0]). For RNNs trained on all the days, adding synthetic data reduced the

character error rate percentage by 2.7 (95% CI = [2.2, 3.3]). A greater performance improvement for single-day RNNs makes sense, as data augmentation is likely to help more when the data is scarcer.

We added the following clarification to the main Methods section:

> Although this method is simplistic in that it assumes that the neural representation of a character is independent of past and future characters, it was nevertheless important ~~This data augmentation step was critical~~ for achieving high performance (decreased the error rate percentage by 12.9 when training on single days and 2.7 when training on all days; SFig. 3A).

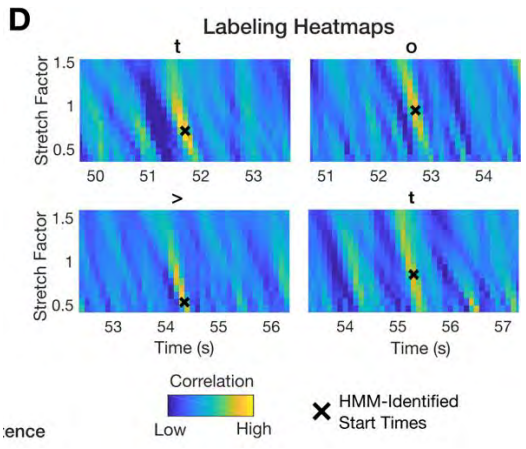
Note that there is no need to label the synthetic sentence with the HMM, since the character identities at each time step are already known. All that is needed is to straight-forwardly construct a time series of probability "targets" that the RNN is trained to output when the synthetic data is given as input. These targets consisted of (1) a one-hot representation of the active character at each time step and (2) a binary "new character" signal which went high whenever a new character started (and remained high for 200 ms). A "one-hot" representation simply means a vector whose entries are all equal to zero except for the entry corresponding to the currently active character (which is equal to one). The following text from the Supplemental Methods defines these target variables in detail:

> Step 6: Construct RNN Targets Finally, target variables for supervised RNN training were generated using the letter start times found above. Two target time series were created: a series of one-hot character vectors (y_t), where each vector is a one-hot representation of the most recently started character, and a scalar time series (z_t) that indicates whether *any* new character has recently been started. The z_t signal allows repeated characters to be distinguished (these would otherwise appear identical to a longer, single character as seen through y_t).
>
> Intuitively, y_t is a 'sample and hold' signal that stores whatever the most recently started character was indefinitely. For example, even if T5 pauses for several seconds after writing the character "a", y_t will still continue to reflect "a" indefinitely until a new character is started. The z_t signal is a complementary binary signal that goes high for a brief time whenever *any* new character begins. z_t can be thresholded to detect the presence of new letters and type them on the screen, which we did online. More formally, y_t and z_t were defined as follows:
>
> $$y_{t,i} = \begin{cases} 0, & \text{the most recently started character was not i} \\ 1, & \text{the most recently started character was i} \end{cases}$$
>
> $$z_t = \begin{cases} 0, & \text{the most recent character was started} > 200 \text{ ms ago} \\ 1, & \text{the most recent character was started} \leq 200 \text{ ms ago} \end{cases}$$

We added extra text to the Methods section to clarify the definition of a "one-hot" representation:

> The vector of target character probabilities (denoted as y_t above) was constructed by setting the probability values at each time step to be a one-hot representation of the most recently started character (i.e., the most recently started character's entry in the vector is equal to 1 while all other entries are 0).

Note that the heatmaps shown in Supplemental Figure 2D (and reproduced below for convenience) were only used to qualitatively assess whether the HMM labeling process succeeded in a reasonable way. The heatmaps themselves were not directly used to construct the RNN targets.



SFig. 2D.

The only thing that was used to construct the RNN targets were the "HMM-identified Start Times", i.e. the time steps when each character began to be written in the training data (as determined by the HMM). Since the heatmaps show hotspots around these HMM-identified start times, we can infer that the labeling process was reasonably accurate (this is just a useful method for sanity checking the labeling). The true proof of the labeling process is the high performance of the RNN decoder that results from using those labels. We added the following disclaimer to the supplemental figure legend:

> Note that these heatmaps are depicted only to qualitatively show label quality and aren't used for training (only the character start times are needed to generate the targets for RNN training).

E. Conclusions

The conclusions are generally based on findings in the work performed in One subject. At times though there are some overstatements about the far reaching ability of the technology which should be scaled down. For example, I did not find the conclusion that this is a BCI without visual feedback to be convincing. If it were, then how can the authors explain the difference in performance between the on-prompt typing and self-paced typing? It is unclear whether there was any type of eye tracking to determine the type of visual feedback the subject was receiving at each moment. For example, was the subject always staring at the text prompt, or was the subject always looking to the decoded characters? Or a combination of both? unless they have an objective measure of visual feedback, it is unclear whether the BCI was truly operating without vision as claimed.

Thank you for these points about visual feedback. As explained above, we have eliminated the claim that the BCI can operate without visual feedback.

F. Suggested improvements:

In addition to the above, I think a critical experiment/analysis to be performed is one in which the authors characterize the longevity and stability of representation of neural signals of the decoded variable(s). The extensive calibration process indicates that the data is highly nonstationary but none of this is characterized.

We thank the reviewer again for these insightful and important suggestions, which we believe have significantly improved the paper by leading us to perform new analyses that demonstrate the feasibility of training decoders with a more limited calibration process. Again, this is all described in detail above.

Based on a few published studies, it is reasonably expected that the implanted device can leverage single cell resolution of neural spiking signals within the first year of implant. However, authors used multiunit activity (binned threshold crossing), implying the activity could not be spike sorted to reveal individual

neuronal activity encoding of the pen tip velocity. More explanation should be provided on how the nonuniform distribution of session dates affected the data quality. Authors explain in the supplementary material that this approach allowed them to "leverage information from more electrodes, since many electrodes recorded activity from multiple neurons that could not be precisely spike-sorted into single units." Although they cite a paper from their group that demonstrated that neural population structure can be accurately estimated from threshold crossing rates alone, it is unclear if sorting spikes from a lower number of electrodes (which they did not state) on which single units could be identified would provide similar results.

Thank you for these questions. First, we would like to clarify that our use of threshold crossings was not motivated by an inability to spike-sort single neuron activity. As stated above, we added a new supplemental figure to demonstrate that high-quality spiking activity can still be recorded on these arrays 1200 days post-implant (reproduced again below for convenience).

Supplemental Figure 6. Example spiking activity recorded from each microelectrode array. (A) Participant T5's MRI-derived brain anatomy. Microelectrode array locations (blue squares) were determined by co-registration of postoperative CT images with preoperative MRI images. (B) Example spike waveforms detected during a ten second time window are plotted for each electrode (data were recorded on post-implant day 1218). Each rectangular panel corresponds to a single electrode and each blue trace is a single spike waveform (2.1 millisecond duration). Spiking events were detected using a -4.5 RMS threshold, thereby excluding almost all background activity. Electrodes with a mean threshold crossing rate ≥ 2 Hz were considered to have 'spiking activity' and are outlined in red (note that this is a conservative estimate that is meant to include only spiking activity that could be from single neurons, as opposed to multiunit 'hash'). Results show that many electrodes still record large spiking waveforms that are well above the noise floor (the y-axis of each panel spans 330 µV, while the background activity has an average RMS value of only 6.4 µV). On this day, 92 electrodes out of 192 had a threshold crossing rate ≥ 2 Hz.

We added the following sentence to the Methods section to give the reader a better understanding of the data quality:

> Note that both arrays still recorded high-quality spiking activity from many electrodes; on average, 81.9 ± 5.6 (mn ± sd) out of 192 electrodes recorded spike waveforms each day at a rate of at least 2 Hz when using a spike-detection threshold of -4.5 RMS (see SFig 6).

Additionally, we now clarify that our decision to use multiunit threshold crossings was not because spike waveforms could no longer be recorded on the arrays:

We used multiunit threshold crossing rates as neural features for analysis and neural decoding (as opposed to spike-sorted single units). This was not because spike waveforms could not be recorded (see SFig 6 for examples); rather, u~~U~~sing multiunit threshold crossings allowed us to leverage information from more electrodes, since many electrodes recorded activity from multiple neurons that could not be precisely spike-sorted into single units.

In our experience, using threshold crossings is simpler, can lead to higher performance (for BCIs), higher signal-to-noise ratios (for neural encoding analyses), and greater stability since action potential waveforms are able to grow/shrink some without affecting threshold crossing detection. Partly though, this depends on how spike-sorted neurons are defined. If one includes *only* well-isolated single neurons, then this excludes a lot of potential data and decreases BCI performance [1]. Good performance can be achieved by spike-sorting more liberally and including multiunit clusters, but this approach does not seem to have clear advantages over multiunit threshold crossings alone [1]. Because threshold crossings have been demonstrated to perform just as well (or within 5% at most) as spike-sorted clusters for BCI applications [1] and for analyzing neural population structure [2], we have chosen to use multiunit threshold crossings throughout our paper. Since these ideas have already been demonstrated in prior work from several nonhuman primate groups and clinical trial groups, we believe it is not necessary to revisit this issue by comparing our multiunit results to spike-sorted results.

[1] Christie, Breanne P., Derek M. Tat, Zachary T. Irwin, Vikash Gilja, Paul Nuyujukian, Justin D. Foster, Stephen I. Ryu, Krishna V. Shenoy, David E. Thompson, and Cynthia A. Chestek. "Comparison of Spike Sorting and Thresholding of Voltage Waveforms for Intracortical Brain–Machine Interface Performance." *Journal of Neural Engineering* 12, no. 1 (December 2014): 016009. https://doi.org/10.1088/1741-2560/12/1/016009.

[2] Trautmann, Eric M., Sergey D. Stavisky, Subhaneil Lahiri, Katherine C. Ames, Matthew T. Kaufman, Daniel J. O'Shea, Saurabh Vyas, et al. "Accurate Estimation of Neural Population Dynamics without Spike Sorting." *Neuron* 103, no. 2 (July 17, 2019): 292-308.e4. https://doi.org/10.1016/j.neuron.2019.05.003.

Finally, the reviewer writes that "More explanation should be provided on how the nonuniform distribution of session dates affected the data quality". Since good BCI performance and/or neural encoding results were achieved on all reported dates, we would propose that data quality is reasonably high throughout. Beyond this, we are unsure what particular question the reviewer might be raising related to the nonuniform distribution of dates, but we believe that we have addressed it above when we provided analyses of how much recalibration is needed depending on the time between sessions. Session dates were nonuniform due to (1) variability in the time needed to analyze data, develop decoding techniques, and prepare experiments and (2) fundamental constraints of the clinical trial, which sometimes preclude regular data collection due to outside demands on the participant and/or unrelated experiments taking priority.

G. References: appropriate credit to previous work?
Mostly relevant and appropriate. The work could benefit from a few more citations that documented the idea of training decoders from 'desired' behavioral templates when overt movements could not be performed.

Thank you for this suggestion. We have added the following references to the Methods section where training velocity decoders to reconstruct pen trajectories is discussed:

As each character was drawn, the X and Y velocity trajectories of the mouse pointer were recorded.~~;~~ T~~t~~hese templates then defined the target velocity vector for the decoder on each time step of each trial, much like prior work has trained decoders to predict the user's 'intended' velocity for continuous movement BCIs [10,11].

[10]Collinger, Jennifer L, Brian Wodlinger, John E Downey, Wei Wang, Elizabeth C Tyler-Kabara, Douglas J Weber, Angus JC McMorland, Meel Velliste, Michael L Boninger, and Andrew B Schwartz. "High-Performance Neuroprosthetic Control by an Individual with Tetraplegia." *The Lancet* 381, no. 9866 (February 2013): 557–64. https://doi.org/10.1016/S0140-6736(12)61816-9.

[11]Gilja, Vikash, Chethan Pandarinath, Christine H. Blabe, Paul Nuyujukian, John D. Simeral, Anish A. Sarma, Brittany L. Sorice, et al. "Clinical Translation of a High-Performance Neural Prosthesis." *Nature Medicine* 21, no. 10 (October 2015): 1142–45. https://doi.org/10.1038/nm.3953.

H. Clarity and context: lucidity of abstract/summary, appropriateness of abstract, introduction and conclusions

No issues.

Thank you.

Again, we deeply appreciate all of these helpful questions and recommendations!

Reviewer Reports on the First Revision:

Referee #2 (Remarks to the Author):

The authors have adequately addressed my concerns and I feel it is suitable for publication. I congratulate them on an impressive work.

Referee #3 (Remarks to the Author):

The revised manuscript has substantially improved in a number of aspects. First, the authors have considerably scaled down some unsubstantiated claims (such as the visual feedback). They also clarified the difference between imagined and attempted movements in their explanation of the results. Second, the authors present new results to address some of my comments. In particular, they clarified that the work is primarily a classification approach of discrete neural activity states as opposed to continuous decoding. They also propose an unsupervised decoder recalibration method using language models that can achieve high performance without interrupting the user. This benefits from the existence of language models to streamline the recalibration process which has been a major element to combat neural signal variability that undoubtedly has an effect on their primary outcome measure: typing speed.

The authors, however, suggested that some of my proposed improvements should be part of future manuscript(s), particularly comments related to longevity of signals affecting decoding reliability and robustness. I think the authors understood my argument in the wrong context -- that their approach should be ready for prime time deployment in clinical applications which was not what I intended. My issue has to do with the level of explanations provided given the results they observed, which I feel is not at the level of the findings and can still be improved. Let me take some space below to clarify what I mean.

The work is primarily a multi-hierarchical classification of neural population dynamic states that starts with non-linear transformation of raw, thresholded activity to create the spatiotemporal templates to be used later for classification (example illustrated nicely in Fig 1E). In the context of online decoding, any features extracted from neural activity will be affected by variability resulting from multiple factors (e.g. array longevity, plasticity in neuronal tuning, attentional levels, etc). The variability can be quantified through two elements: 1) signaling – which has to do with the quality of spikes and robustness of sorting to permit reliable extraction of firing rates from as many single units. This was not quantified in this work because they did not do spike sorting (see comment above). And 2) information coding – which has to do with the actual representation of characters being encoded in the neural activity. Again, this was not characterized in this work because the authors stated that this is out of the manuscript focus and should be the topic of a future manuscript.

Interestingly, the authors demonstrate in new Supp Fig 4 how much of this variability resulted in variations in the decoded spatiotemporal templates. In particular, the shrinkage effect that the new figure shows highlights the main issue that I have raised. Somehow this information is embedded in the 'new' knowledge that the RNN learns with continued additions of new templates to the training dataset. However, there should be more in depth explanation or discussion on how this observation (as well as the 7-day stability result they also found, see related comment below) could enhance our understanding of handwriting movement representation in the brain. At the least, more discussion should be included regarding how decoders should be engineered to account for movements that have similar structured temporal variations (which could be very useful for other types of sequential movements). While it is not a major flaw and the modified text helps, I think the authors need to improve the discussion related to these two points in particular.

The new result in Figure 3 shows offline decoding performance when less than 50 calibration

sentences were used. While the result is interesting, the authors need to put it into perspective given that online decoding performance does diminish considerably compared to offline simulations, as their own results in Table 1 have shown. How would this result carry over to the online decoding case? how this performance is a function of character probability in these sentences, as well as the particular choice of the 10 sentences?
Another related issue is the explanations given to the difference between online and offline decoding performance. For example, it is well established that well isolated unit spiking does provide more information compared to local field potentials for BCI decoding but comes at the cost of increased computational complexity and variability over time, both within session and across sessions. I did not find their argument about not using spike sorting to be particularly compelling. Even though it is a subjective process as they state – but so is the thresholding process they've used, it has the potential to increase their information rate and consequently typing speed which is their main outcome measure. Unless it was observed that this process does somehow affect the spatiotemporal templates they use in the classification, the reasons for not using putative single or multi-unit clusters of waveforms in building the firing rate templates are not entirely clear.

It is also important to explain why 7 days or less seem to maintain uncalibrated decoder accuracy. It is important to cite prior published work in which it was demonstrated that the same duration tends to also be associated with stability of single unit spiking1,2. Is this a coincidence? I think the same is happening here, that decoders need to be calibrated because unit spiking and character representation seems to shift over intervals > 7 days.
1. Dickey, Adam S., et al. "Single-unit stability using chronically implanted multielectrode arrays." Journal of neurophysiology 102.2 (2009): 1331-1339.
2. Eleryan, Ahmed, et al. "Tracking single units in chronic, large scale, neural recordings for brain machine interface applications." Frontiers in neuroengineering 7 (2014): 23.

Specific comments:
Authors state that "The character templates were made by drawing each character with a computer mouse in the same way as T5 described writing the character. As each character was drawn, the X and Y velocity trajectories of the mouse pointer were recorded.;" I can understand how T5 can describe how the final shape would look like, but it's unclear how can T5 describe the velocity by which he attempted to write different parts of the characters from which the target velocity vector for the decoder was defined.
Line 115: How did neural activity look like when an error was made? and when the subject was provided visual feedback about the language model's autocorrection of that error? Did the subject stop modulating, eventually relying on the model to autocorrect, or did he continue to modulate neural activity to correct the typo? Was the decoder 'disengaged' in those instances? did the neural activity occupy different regions of the state space relative to the intended character or the corrected character?
Authors report that T5 spent 93% of the time looking at the prompt in the copy-typing task. They should also state what the eye tracking statistics were in the free typing trials in which there was no prompt. They also did not respond to the question if the perception of errors had any influence on the decoded patterns, particularly given that they cite their own work showing how errors result in distinct signature in motor cortex. They should clarify if and how these signatures, particularly when an error was made in the free typing trials, was handled by the decoder.

Authors have responded to my comment about the toy example by stating that the example included noise. My reference was to Figure 3F in which noise is absent in the trajectories shown. While I agree with the authors that "as the amount of noise increases, complex trajectories (may) become easier to classify because their nearest neighbor distances are larger (and thus nearby trajectories are less likely to be ", this is the case only under two assumptions: 1) The noise is white (as they have simulated already) and 2) the signal (i.e. the trajectory being classified) is uncorrelated with that noise. In the brain, however, there is ample evidence to suggest that the noise is not white, and is strongly correlated with the signal, and if not, it would at least be correlated among adjacent electrodes.

As they show in their inequality, when the distance between noisy fa and fb is less than the noise norm, misclassification will happen. However, if fa is more different than fb but the noise is more similar (i.e. correlated) to fb, adding that noise to fa will bring the noisy fa closer to fb, thus increasing the probability of misclassification. This is the case when the noise cluster extends along specific directions closer to those spanned by the signals. The examples simulated in the rebuttal use temporally colored noise, but the extent to which it is correlated with the actual signals being classified is unclear. I suggest that the authors bring more realism into their toy example regarding the noise characteristics (especially its temporal correlation with the signal while keeping its variance small) to make the explanation they are offering more compelling.

Overall, I think the work is very valuable and would be an important contribution to the field, provided the authors address some of the remaining issues above.

Author Rebuttals to First Revision:

Reply to Reviewers – Round 2

Note: reviewers' comments appear in **black text**. Our replies appear in **blue text**, and revised manuscript text appears indented (with old text shown in **black** and new edits in **red**).

Referee #3 (Remarks to the Author):

The revised manuscript has substantially improved in a number of aspects. First, the authors have considerably scaled down some unsubstantiated claims (such as the visual feedback). They also clarified the difference between imagined and attempted movements in their explanation of the results. Second, the authors present new results to address some of my comments. In particular, they clarified that the work is primarily a classification approach of discrete neural activity states as opposed to continuous decoding. They also propose an unsupervised decoder recalibration method using language models that can achieve high performance without interrupting the user. This benefits from the existence of language models to streamline the recalibration process which has been a major element to combat neural signal variability that undoubtedly has an effect on their primary outcome measure: typing speed.

Thank you for this kind summary and recognition of the effort we devoted to address your helpful questions and improve the manuscript. We are also grateful for the additional suggestions detailed below. We have done our best to address them within the strict space-constraints of a Nature article, which restricts the main text to 6.0 pages (manuscript length before this final revision was 6.7 pages, due to the inclusion of many excellent requests by all three reviewers).

The authors, however, suggested that some of my proposed improvements should be part of future manuscript(s), particularly comments related to longevity of signals affecting decoding reliability and robustness. I think the authors understood my argument in the wrong context -- that their approach should be ready for prime time deployment in clinical applications which was not what I intended. My issue has to do with the level of explanations provided given the results they observed, which I feel is not at the level of the findings and can still be improved. Let me take some space below to clarify what I mean.

Thank you for these helpful clarifications.

The work is primarily a multi-hierarchical classification of neural population dynamic states that starts with non-linear transformation of raw, thresholded activity to create the spatiotemporal templates to be used later for classification (example illustrated nicely in Fig 1E). In the context of online decoding, any features extracted from neural activity will be affected by variability resulting from multiple factors (e.g. array longevity, plasticity in neuronal tuning, attentional levels, etc). The variability can be quantified through two elements: 1) signaling – which has to do with the quality of spikes and robustness of sorting to permit reliable extraction of firing rates from as many single units. This was not quantified in this work because they did not do spike sorting (see comment above). And 2) information coding – which has to do with the actual representation of characters being encoded in the neural activity. Again, this was not characterized in this work because the authors stated that this is out of the manuscript focus and should be the topic of a future manuscript.

Thank you for this clarification. It is true that characterizing single neuron spiking quality and/or the neural representation of handwriting are not the focus of this work (although we do think Fig. 1 provides some important characterization of the neural representation of handwriting, by showing that pen-tip velocity can be decoded and by providing a low-dimensional visualization of the neural population structure via t-SNE).

Interestingly, the authors demonstrate in new Supp Fig 4 how much of this variability resulted in variations in the decoded spatiotemporal templates. In particular, the shrinkage effect that the new figure shows highlights the main issue that I have raised. Somehow this information is embedded in the 'new' knowledge that the RNN learns with continued additions of new templates to the training dataset. However, there should be more in depth explanation or discussion on how this observation (as well as the 7-day stability result they also found, see related comment below) could enhance our understanding of handwriting movement representation in the brain.

Thank you for this suggestion. As the reviewer mentions below, one likely reason for the changes in multiunit neural activity we observed over time (including the shrinkage effect) is the instability of spiking activity as observed through microelectrode arrays, an unknown fraction of which is caused by device micromotion. Therefore, we think that any changes in multiunit neural activity over time do not necessarily provide insight into neural plasticity or neural representations, as an unknown portion of that change is due to array micromotion. In the main text, when discussing decoder retraining, we now clarify for readers that the source of the neural changes may be due to plasticity or device micromotion:

> Retraining helps account for changes in neural recordings that accrue over time (which might be caused by neural plasticity or electrode array micromotion).

At the least, more discussion should be included regarding how decoders should be engineered to account for movements that have similar structured temporal variations (which could be very useful for other types of sequential movements). While it is not a major flaw and the modified text helps, I think the authors need to improve the discussion related to these two points in particular.

We appreciate this suggestion. The new Results section added in the previous revision highlights extensively the fact that neural decoders are negatively affected by changes in neural activity over time and typically require frequent retraining to combat this (either with explicit calibration data or via unsupervised retraining). After reporting our new analyses on this point, we offer the following interpretation for how decoders might be designed to be robust to temporal variations that have the medium-length time scale shown in our analyses:

> The above results are promising for clinical viability, as they suggest that unsupervised decoder retraining, combined with more limited supervised retraining after longer periods of inactivity, may be sufficient to achieve high performance.

While we too see the value in discussing this issue more thoroughly, particularly with regards to the interesting temporal shrinkage effect now shown in Extended Data Figure 4, the strict space

constraints of a Nature article prevent us from doing so (without removing other central results or discussion points). We have discussed this, and other space limitation restraints with the Editor to be sure that we are balancing this appropriately.

The new result in Figure 3 shows offline decoding performance when less than 50 calibration sentences were used. While the result is interesting, the authors need to put it into perspective given that online decoding performance does diminish considerably compared to offline simulations, as their own results in Table 1 have shown. How would this result carry over to the online decoding case? how this performance is a function of character probability in these sentences, as well as the particular choice of the 10 sentences?

Thank you for these suggestions. We now more explicitly highlight that these new analyses were performed offline and thus require future work to confirm online:

> … unsupervised decoder retraining, combined with more limited supervised retraining after longer periods of inactivity, may be sufficient to achieve high performance. Nevertheless, future work must confirm this online, as offline simulations are not always predictive of online performance.

While we do appreciate that decoders can perform worse online than they do offline, we do not think this is always the case (nor necessarily to be expected). Since in this work the user only receives delayed feedback of the decoded characters after they have been completed/detected by the RNN, we think offline simulations are more likely to transfer to the online case as compared to continuous motion BCIs which rely heavily on moment-to-moment visual feedback corrections.

To clarify, note that the results in Table 1 do not show a failure of decoding results to transfer to the online domain. Although the last row of the table reports best performance with an offline decoder, this offline decoder was *acausal* (a bidirectional RNN) and was not tested online or shown to be worse online. Its high performance was likely due to its acausal nature. Note that this acausal decoder was tested to provide a point of comparison to prior BCI work which has also used acausal methods.

Finally, we would like to clarify that the 10 sentences selected for calibration in our offline simulation were subsampled at even intervals from the 50 possible sentences (thus ensuring that the 10 sentences are distributed evenly in time). To understand the effect of this choice on performance, we re-ran the analysis 10 more times, each time with sentences chosen at random (i.e., uniformly at random instead of deterministically at even intervals). Results show a tight clustering near the originally reported result, suggesting that the choice of sentences does not have a strong effect on decoder performance. The originally reported error rate was 8.5%; the mean of these new random runs was 9.2% with a standard deviation of 0.6%. In the Methods, we now clarify that, in the offline simulations shown in Figure 3, sentences were subsampled from the original set of sentences at even intervals and that this choice does not affect the conclusions:

> When reducing the amount of calibration data, we subsampled from the original 50 sentences at even intervals (thus ensuring that the subsampled data contained sentences spaced evenly in

time). Note that results are similar when choosing sentences uniformly at random. To test this, we re-ran the analysis 10 more times using 10 sentences chosen randomly instead of evenly. The reported error rate in Fig. 3a was 8.5% for 10 sentences; the mean of these 10 random runs was 9.2% with a standard deviation of 0.6%.

Another related issue is the explanations given to the difference between online and offline decoding performance. For example, it is well established that well isolated unit spiking does provide more information compared to local field potentials for BCI decoding but comes at the cost of increased computational complexity and variability over time, both within session and across sessions. I did not find their argument about not using spike sorting to be particularly compelling. Even though it is a subjective process as they state – but so is the thresholding process they've used, it has the potential to increase their information rate and consequently typing speed which is their main outcome measure. Unless it was observed that this process does somehow affect the spatiotemporal templates they use in the classification, the reasons for not using putative single or multi-unit clusters of waveforms in building the firing rate templates are not entirely clear.

Thank you for these considerations. Ultimately, since we did not compare multiunit threshold crossings to spike-sorted clusters in this work, it is unknown whether spike-sorting could have improved our system's performance. It does seem plausible that at least some small performance benefit could have been gained by using spike-sorting. The only place in the manuscript where this issue is addressed is a paragraph in the Methods that motivates our choice of multiunit threshold crossings. We have revised this paragraph as follows:

> We used multiunit threshold crossing rates as neural features for analysis and neural decoding (as opposed to spike-sorted single units). We made this choice to simplify the methods, not ~~This was not~~ because spike waveforms could not be recorded (see ~~SFig 6~~Extended Data Fig. 7 for examples~~); ~~). ~~rather, using multiunit threshold crossings allowed us to leverage information from more electrodes, since many electrodes recorded activity from multiple neurons that could not be precisely spike-sorted into single units.~~ Recent results ~~indicate~~ suggest that neural population structure can be accurately estimated from threshold crossing rates alone [45](Trautmann et al., 2019), and that neural decoding performance is ~~similar~~ comparable (within 5%) to using sorted units~~-~~ (Chestek et al., 2011; Christie et al., 2014) – although see also (Todorova et al., 2014)[46].

It is also important to explain why 7 days or less seem to maintain uncalibrated decoder accuracy. It is important to cite prior published work in which it was demonstrated that the same duration tends to also be associated with stability of single unit spiking1,2. Is this a coincidence? I think the same is happening here, that decoders need to be calibrated because unit spiking and character representation seems to shift over intervals > 7 days.

1. Dickey, Adam S., et al. "Single-unit stability using chronically implanted multielectrode arrays." Journal of neurophysiology 102.2 (2009): 1331-1339.
2. Eleryan, Ahmed, et al. "Tracking single units in chronic, large scale, neural recordings for brain machine interface applications." Frontiers in neuroengineering 7 (2014): 23.

Thank you for this suggestion. Indeed, a relatively high stability within a 7-day window is consistent with this prior work, which we now cite in the main text:

We found that when only 2-7 days passed between sessions, performance was reasonable with *no* decoder retraining (11.1% raw error rate, 1.5% with a language model), as might be expected from prior work indicating short-term stability of neural recordings[19–21].

19. Dickey, A. S., Suminski, A., Amit, Y. & Hatsopoulos, N. G. Single-Unit Stability Using Chronically Implanted Multielectrode Arrays. *J Neurophysiol* **102**, 1331–1339 (2009).
20. Eleryan, A. *et al.* Tracking single units in chronic, large scale, neural recordings for brain machine interface applications. *Front. Neuroeng.* **7**, (2014).
21. Downey, J. E., Schwed, N., Chase, S. M., Schwartz, A. B. & Collinger, J. L. Intracortical recording stability in human brain–computer interface users. *J. Neural Eng.* **15**, 046016 (2018).

Specific comments:
Authors state that "The character templates were made by drawing each character with a computer mouse in the same way as T5 described writing the character. As each character was drawn, the X and Y velocity trajectories of the mouse pointer were recorded.;" I can understand how T5 can describe how the final shape would look like, but it's unclear how can T5 describe the velocity by which he attempted to write different parts of the characters from which the target velocity vector for the decoder was defined.

To clarify, the target velocity vectors were only meant to be a rough approximation of T5's intended movement, based on the assumption that another person drawing the same character shape with a computer mouse would naturally follow a similar velocity trajectory. T5 never described the velocity of each character, and no attempt was made to check it against T5's understanding.

The algorithm we used to decode the pen tip velocity is invariant to linear scaling in the overall writing speed (since it stretches/shrinks each target template to best match the neural activity), but is not invariant to more subtle differences in the velocity of each individual stroke. Nevertheless, the fact that recognizable character trajectories could be decoded with this method shows that, even though the computer mouse trajectories are (necessarily) only rough approximations of T5's intended velocities, they are close enough to enable successful decoder training.

We now further clarify this point in the Methods section:

To train the decoder, we used hand-made templates that describe each character's pen trajectory. The character templates were made by drawing each character with a computer mouse in the same way as T5 described writing the character. As each character was drawn, the X and Y velocity trajectories of the mouse pointer were recorded. These templates then defined the target velocity vector for the decoder on each time step of each trial, much like prior work has trained decoders to predict the user's "intended" velocity for continuous movement tasks [2,49]. These templates were only intended to be a rough approximation of T5's intended pen tip velocities, based on the assumption that another person drawing the same character shape with a computer mouse would naturally follow a similar velocity trajectory (up to some time-scaling factor, to account for differences in overall writing speed).

Line 115: How did neural activity look like when an error was made? and when the subject was provided visual feedback about the language model's autocorrection of that error? Did the subject stop modulating, eventually relying on the model to autocorrect, or did he continue to modulate neural activity to correct the typo? Was the decoder 'disengaged' in those instances? did the neural activity occupy different regions of the state space relative to the intended character or the corrected character?

Thank you for these interesting suggestions. We would like to clarify that the language model was applied *offline* only, in a retrospective analysis to simulate an autocorrect feature. Thus, the participant never saw the autocorrections. To make this clearer, we added the word "Offline" to the system diagram in Figure 2 that depicts the language model (previously it was described only as "Retrospective" in the figure diagram).

Authors report that T5 spent 93% of the time looking at the prompt in the copy-typing task. They should also state what the eye tracking statistics were in the free typing trials in which there was no prompt. They also did not respond to the question if the perception of errors had any influence on the decoded patterns, particularly given that they cite their own work showing how errors result in distinct signature in motor cortex. They should clarify if and how these signatures, particularly when an error was made in the free typing trials, was handled by the decoder.

Thank you again for these suggestions. As the space limitations of a Nature article are strict, we feel we must retain the manuscript's focus on the central results – (1) demonstrating that handwriting movements are neurally encoded even years after paralysis, (2) that complete handwritten sentences can be neurally decoded at high speeds and accuracies using a novel decoding approach, and (3) that theoretical considerations suggest that temporally complex movements are easier to decode than point-to-point movements, making high-performance handwriting decoding possible.

While the effect of errors on neural activity (and, relatedly, the pattern of gaze in BCI use) is an important and interesting topic, which we too are deeply curious about, the results of such analyses would not directly bear on these central claims. Even if they did, there would unfortunately be no space to include or discuss them in the main text without removing other central results.

To clarify, our decoder was not designed to detect neural signatures of error – it was trained only to maximize classification accuracy. Thus, errors were not handled in a special way by the decoder. If it made an error, it simply continued unaware as if it had not.

Finally, we would like to also clarify that our prior studies on the encoding of errors are not cited in this manuscript – we cited them only in our response to reviewers in the previous round.

Authors have responded to my comment about the toy example by stating that the example included noise. My reference was to Figure 3F in which noise is absent in the trajectories shown. While I agree with the authors that "as the amount of noise increases, complex trajectories (may) become easier to classify because their nearest neighbor distances are larger (and thus nearby trajectories are less likely to be ", this is the case only under two assumptions:

1) The noise is white (as they have simulated already) and 2) the signal (i.e. the trajectory being classified) is uncorrelated with that noise. In the brain, however, there is ample evidence to suggest that the noise is not white, and is strongly correlated with the signal, and if not, it would at least be correlated among adjacent electrodes.

As they show in their inequality, when the distance between noisy fa and fb is less than the noise norm, misclassification will happen. However, if fa is more different than fb but the noise is more similar (i.e. correlated) to fb, adding that noise to fa will bring the noisy fa closer to fb, thus increasing the probability of misclassification. This is the case when the noise cluster extends along specific directions closer to those spanned by the signals. The examples simulated in the rebuttal use temporally colored noise, but the extent to which it is correlated with the actual signals being classified is unclear. I suggest that the authors bring more realism into their toy example regarding the noise characteristics (especially its temporal correlation with the signal while keeping its variance small) to make the explanation they are offering more compelling.

Thank you for this helpful suggestion. In addition to temporally correlated noise, we now also simulated signal-correlated noise (i.e., noise that spans the dimensions which connect the class means). The results continue to hold in the case of signal-spanning noise as well. This can be explained by the fact that signal-spanning noise acts like white noise in dimensions that span the class means, but is zero elsewhere. Since noise in dimensions that *don't* align with the class means are not as relevant for classification performance, it makes sense that their absence does not change the main result.

We now summarize these new analyses in a new Extended Data Figure (now Extended Data Fig. 5) and Supplementary Note, which we reproduce below at the end of this document for convenience. Additionally, we now call out this Supplementary Note and the assumption of uncorrelated white noise in the main text:

> Although neural noise in the toy model was assumed to be independent white noise, we also found that these results hold for noise that is correlated across time and neurons (Extended Data Fig. 5, Supplementary Note 1).

Thank you again, as we too feel that exploring these additional noise types helps to strengthen the argument.

Overall, I think the work is very valuable and would be an important contribution to the field, provided the authors address some of the remaining issues above.

Thank you again for all of these helpful comments and suggestions. We did our best to incorporate them, given the page limit constraints of a Nature article.

Supplementary Note 1 – Effect of Noise Correlations on the Toy Model of Classifiability

In the toy example presented in Fig. 4F-H, we showed that additional temporal dimensions can be used to improve the classifiability of a set of neural patterns in the presence of Gaussian *white noise* that is uncorrelated across time points and neurons. Under these assumptions, the Euclidean distance between each pair of neural patterns is the relevant factor determining classification accuracy, and it therefore follows that greater temporal dimensionality will improve classification performance if it helps to spread out those patterns more evenly. Here, we examine how *correlated noise* might affect this result.

First, it is helpful to define some terms. Let f_x be a vector that describes the underlying neural trajectory for movement x (i.e., the mean neural firing rates across time for movement x). Each entry in the vector f_x is the mean firing rate for a single time step. To describe multiple neurons, the activity profile of each neuron can be stacked one on top of the other in the vector. Let ε be a neural noise vector of the same length that has a multivariate normal distribution with zero mean and covariance matrix Σ. If Σ is non-diagonal, the noise is said to be correlated.

Given a vector of noisy observed firing rates $r = f_x + \varepsilon$, a maximum likelihood classifier will choose to classify r into the class that has the minimum Mahalanobis distance to r (assuming uniform class priors). In other words:

$$\underset{x}{\operatorname{argmin}}(r - f_x)^T \Sigma^{-1} (r - f_x)$$

In the case of white noise, Σ is a diagonal matrix with all diagonal entries equal to σ. In this case, the classifier will simply choose the class whose mean has the smallest Euclidean distance to r. This justifies the idea that nearest neighbor distances should be increased to reduce classifier confusions (potentially via spreading the neural patterns out into additional temporal dimensions).

If Σ is non-diagonal, this means that the noise cloud will extend more in some directions and less in others. The directions that are most harmful for classification are those that connect nearby class means (e.g., the direction f_x-f_y, as this would make noise more likely to 'corrupt' class x to look more like y). In the general case where Σ can take any arbitrary shape, it is not always true that classification accuracy can be improved by using extra temporal dimensions to increase Euclidean distances. For example, it could be the case that these extra temporal dimensions are particularly noisy, cancelling out the benefit of increased distance between the class means. Nevertheless, under reasonable constructions of Σ that we test below, we show that the toy model in Fig. 4 still holds in the presence of correlated noise.

Temporally Correlated Noise

First, we tested noise with *temporal* correlations (meaning that the noise associated with each neuron was positively correlated in time). This noise can describe slow (but random) fluctuations in neural firing rates over time, and in this sense is more realistic than white noise. Temporal correlations would generally cause the noise to be more concentrated along dimensions that span the class means, since the underlying neural patterns are also smooth across time (as is the case in this toy example). Extended Data Fig. 5a shows examples of temporally correlated noise vectors and the covariance matrix used to generate them. The wide diagonal band in the covariance matrix causes nearby time steps to have correlated noise.

In Extended Data Fig. 5b, we compared the classification accuracy between time-varying trajectories and constant trajectories in the presence of temporally correlated noise, finding an even more pronounced improvement for time-varying trajectories. This is because neural patterns that vary more quickly in time are less aligned with slow-varying noise directions, enabling greater robustness to this type of noise. Here, classification was performed with a maximum likelihood classifier (under the assumption that the means of each class and the covariance matrix of the noise are known). However, results also hold using

a simpler “Euclidean distance” classifier that assumes the noise is white by choosing the class whose mean has the smallest Euclidean distance to *r*.

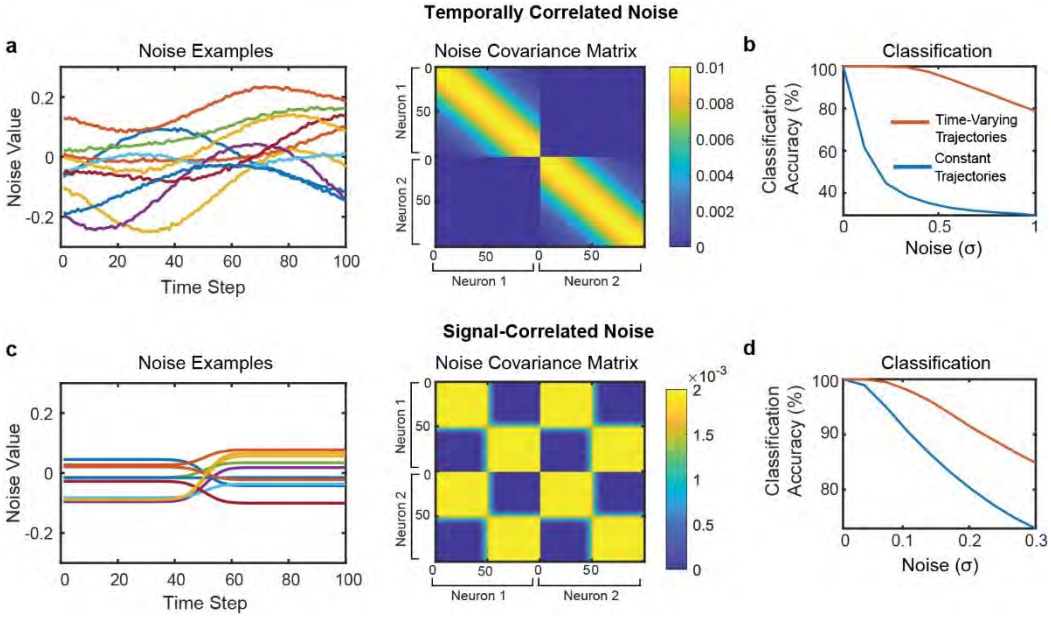
Signal-Correlated Noise

Finally, we tested noise vectors that were directly correlated with the underlying neural signal (that is, noise vectors that contained variance *only* in signal-spanning dimensions that connect the class means, such as f_x - f_y). This type of noise is more realistic than white noise in the sense that neural variability is often larger in neural dimensions that carry the signal. To find these dimensions, PCA was applied separately to the constant and time-varying trajectories to find the one (constant) or two (time-varying) spatiotemporal axes containing the neural signal. The covariance matrix was then designed to place noise in these axes only (with equal variance for each axis):

$$\Sigma = \sigma \mathrm{A}\mathrm{A}^T$$

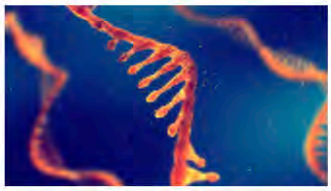
Here, A is a matrix whose columns are the PCA axes and σ scales the overall size of the noise. Extended Data Fig. 5c shows what these noise vectors look like for the time-varying trajectories. Because the time-varying trajectories have only two temporal dimensions, the noise vectors also have this structure (where the first 50 time points are highly correlated with each other and the last 50 time points are highly correlated with each other).

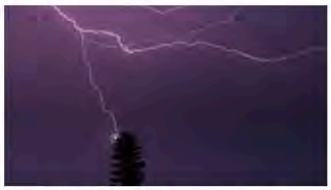
Again, even in the presence of noise that is correlated with the signal, we found that it is still easier to classify time-varying trajectories than constant trajectories (Extended Data Fig. 5d). This result can be explained by the fact that signal-spanning noise acts like white noise in dimensions that span the class means, but is zero elsewhere. Since noise in dimensions that *don’t* align with the class means are not as relevant for classification performance, it makes sense that their absence does not change the main result.



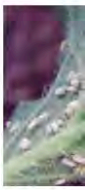
Extended Data Fig. 5: Effect of correlated noise on the toy model of temporal dimensionality. a, Example noise vectors and covariance matrix for temporally correlated noise. On the left, example noise vectors are plotted (each line depicts a single example). Noise vectors are shown for all 100 time steps of neuron 1. On the right, the covariance matrix used to generate temporally correlated noise is plotted (dimensions = 200 x 200). The first 100 time steps describe neuron 1’s noise and the last 100 time steps describe neuron 2’s noise. The diagonal band creates noise that is temporally correlated within each

simulated neuron (but the two neurons are uncorrelated with each other). **b,** Classification accuracy when using a maximum likelihood classifier to classify between all four possible trajectories in the presence of temporally correlated noise. Even in the presence of temporally correlated noise, the time-varying trajectories are still much easier to classify. **c,** Example noise vectors and noise covariance matrix for noise that is correlated with the signal (i.e., noise that is concentrated only in spatiotemporal dimensions that span the class means). Unlike the temporally correlated noise, this covariance matrix generates *spatiotemporal* noise that has correlations between time steps *and* neurons. **d,** Classification accuracy in the presence of signal-correlated noise. Again, time-varying trajectories are easier to classify than constant trajectories.

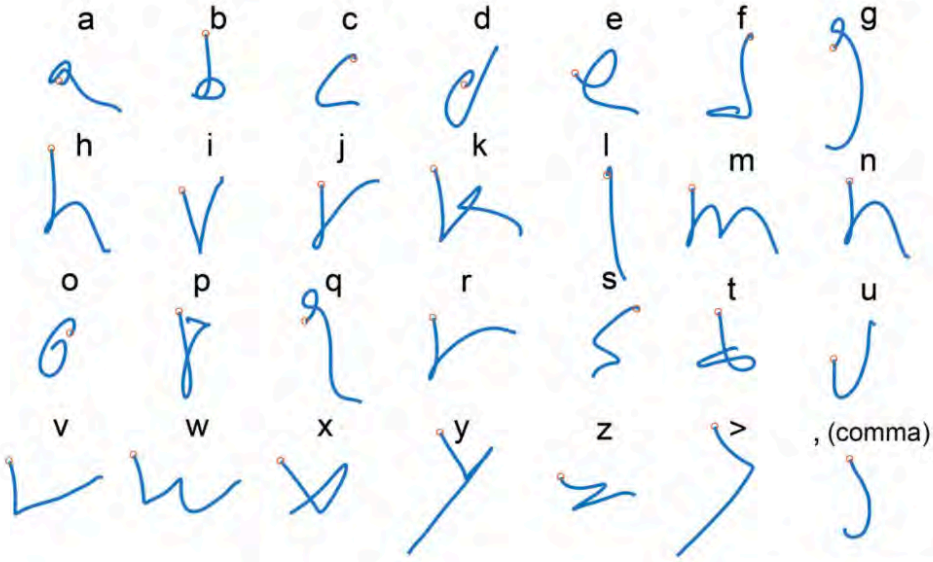

Sugars spice up RNA


Climate change is triggering more lightning strikes in the Arctic


Tropical forest destruction increases, despite the pandemic


Pesticide transferr

SHARE



F. WILLETT *ET AL./NATURE* 2021

Paralyzed person types at record speed—by imagining handwriting

By Kelly Servick | May. 13, 2021 , 4:40 PM

A new approach to brain-computer interface has allowed a paralyzed person to **type with unprecedented speed**, *The Scientist* reported yesterday. As *Science* first reported in 2019, researchers used electrodes implanted in a motor region of the brain **to read out letters** as a person paralyzed from the neck down imagined writing by hand. Previous systems, where users select on-screen letters by moving a cursor with their minds, have reached speeds up to about 40 characters per minute, but the new approach allowed speeds of up to **90 characters per minute with 94% accuracy**, the researchers reported this week in *Nature*. Future improvements to the setup—such as making it smaller, wireless, and easier to calibrate—could ready it for wider clinical use.

***Correction, 14 May, 4:30 p.m.:** A previous version of this story stated that the new approach allowed speeds of up to 90 words per minute. This has been corrected.*

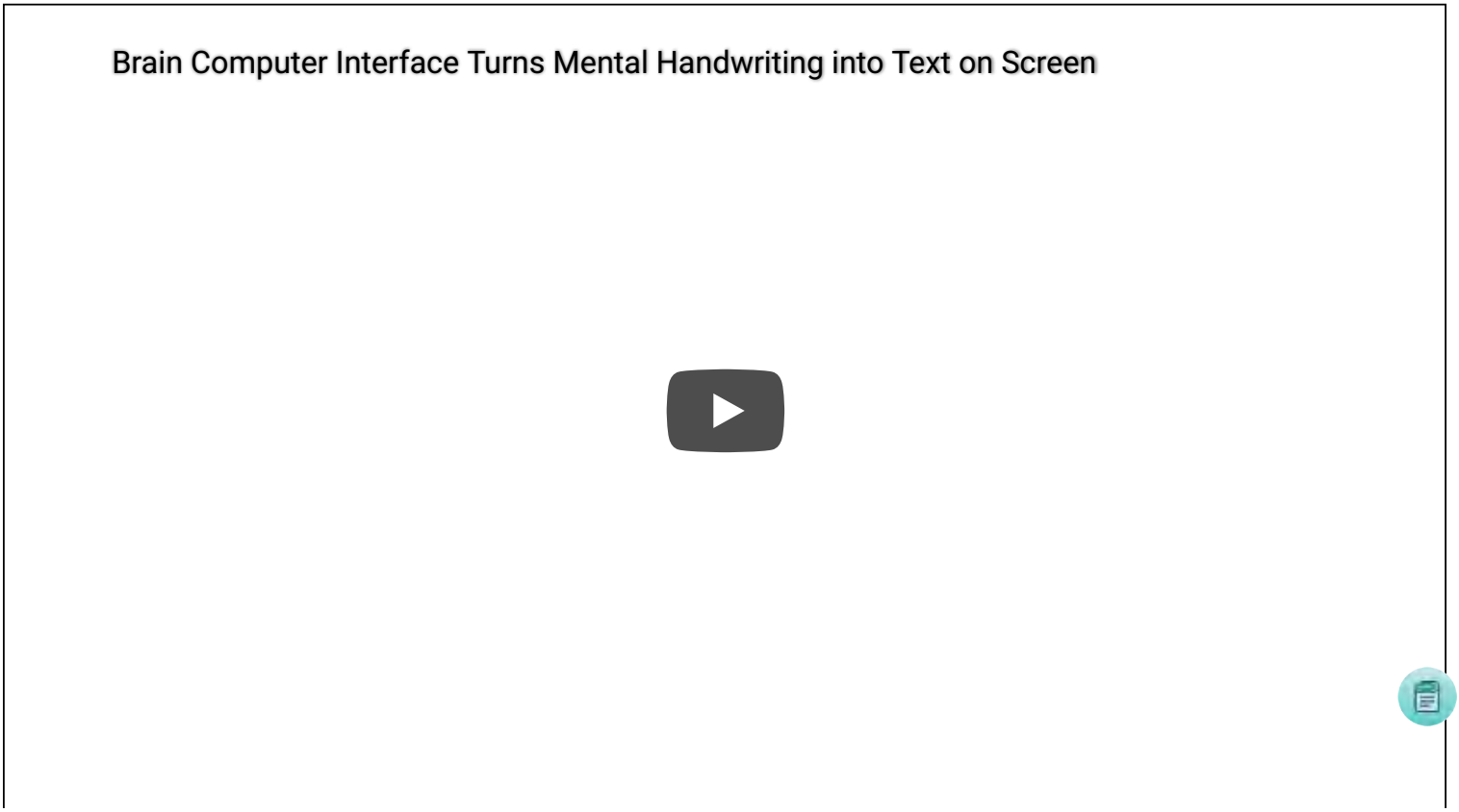
hhmi

MAY 12 2021 Research

Brain Computer Interface Turns Mental Handwriting into Text on Screen

Summary

Researchers have, for the first time, decoded the neural signals associated with writing letters, then displayed typed versions of these letters in real time. They hope their invention could one day help people with paralysis communicate.



Scientists are exploring a number of ways for people with disabilities to communicate with their thoughts. The newest and fastest turns back to a vintage means for expressing oneself: handwriting.

For the first time, researchers have deciphered the brain activity associated with trying to write letters by hand. Working with a participant with paralysis who has sensors implanted in his brain, the team used an algorithm to identify letters as he attempted to write them. Then, the system displayed the text on a screen – in real time.

The innovation could, with further development, let people with paralysis rapidly type without using their hands, says study coauthor Krishna Shenoy, a Howard Hughes Medical Institute Investigator at Stanford University who jointly supervised the work with Jaimie Henderson, a Stanford neurosurgeon.

By attempting handwriting, the study participant typed 90 characters per minute – more than double the previous record for typing with such a "brain-computer interface," Shenoy and his colleagues report in the journal *Nature* on May 12, 2021.

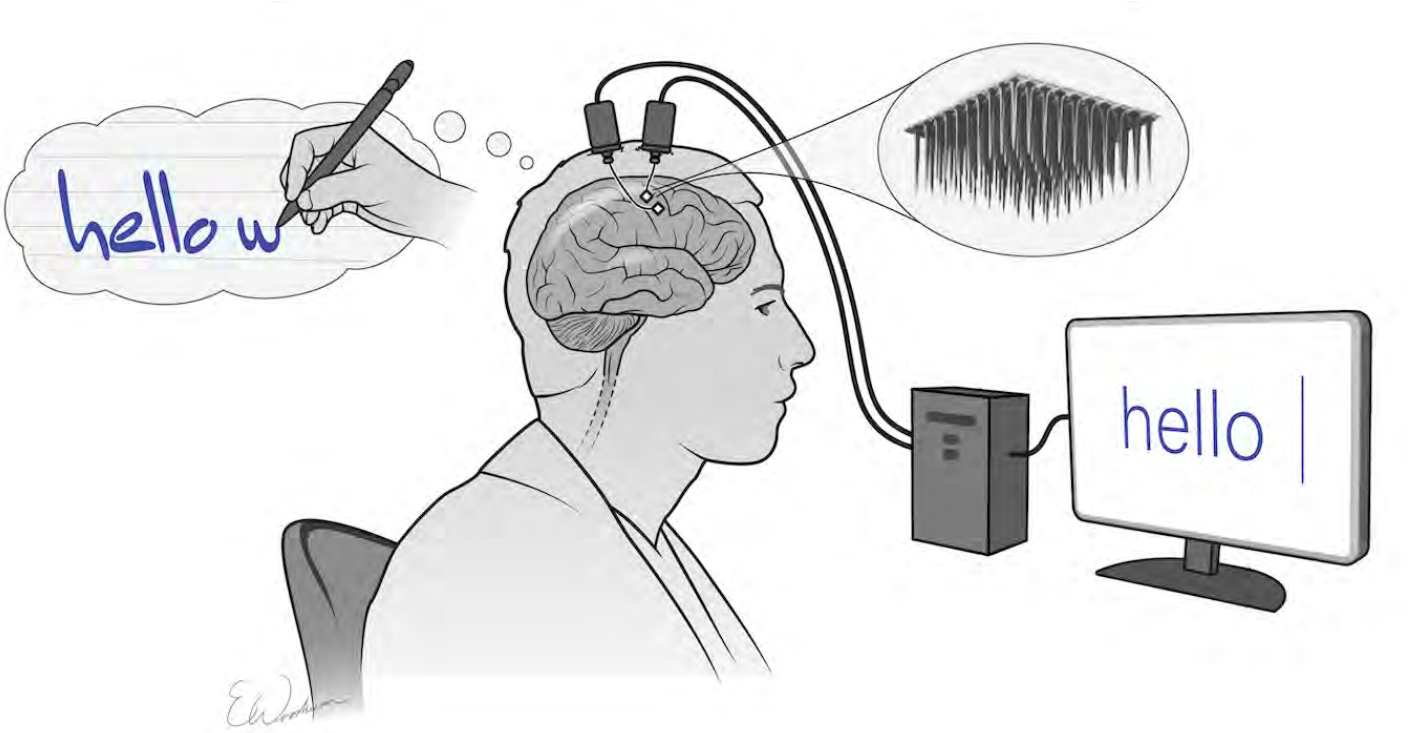
This technology and others like it have the potential to help people with all sorts of disabilities, says Jose Carmena, a neural engineer at the University of California, Berkeley, who was not involved in the study. Though the findings are preliminary, he says, "it's a big advancement in the field."

Brain-computer interfaces convert thought into action, Carmena says. "This paper is a perfect example: the interface decodes the thought of writing and produces the action."

Thought-powered communication

When an injury or disease robs a person of the ability to move, the brain's neural activity for walking, grabbing a cup of coffee, or speaking a sentence remain. Researchers can tap into this activity to help people with paralysis or amputations regain lost abilities.

The need varies with the nature of the disability. Some people who have lost the use of their hands can still use a computer with speech recognition and other software. For those who have difficulty speaking, scientists have been developing other ways to help people communicate.



Two tiny arrays of implanted electrodes relayed information from the brain area that controls the hands and arms to an algorithm, which translated it into letters that appeared on a screen. Credit: F. Willett et al./*Nature* 2021/Erika Woodrum

In recent years, Shenoy's team has decoded the neural activity associated with speech in the hopes of reproducing it. They have also devised a way for participants with implanted sensors to use their thoughts associated with attempted arm movements to move a cursor on a screen. Pointing at and clicking on letters in this way let people type about 40 characters per minute, the previous speed record for typing with a brain computer interface (BCI).

No one, however, had looked at handwriting. Frank Willett, an HHMI research specialist and neuroscientist in Shenoy's group, wondered if it might be possible to harness the brain signals evoked by putting pen to paper. "We want to find new ways of letting people communicate faster," he says. He was also motivated by the opportunity to try something different.

The team worked with a participant enrolled in a clinical trial called BrainGate2, which is testing the safety of BCIs that relay information directly from a participant's brain to a computer. (The trial's director is Leigh Hochberg, a neurologist and neuroscientist at Massachusetts General Hospital, Brown University, and the Providence VA Medical Center.) Henderson implanted two tiny sensors into the part of the brain that controls the hand and arm, making it possible for the person to, for example, move a robotic arm or a cursor on a screen by attempting to move their own paralyzed arm.

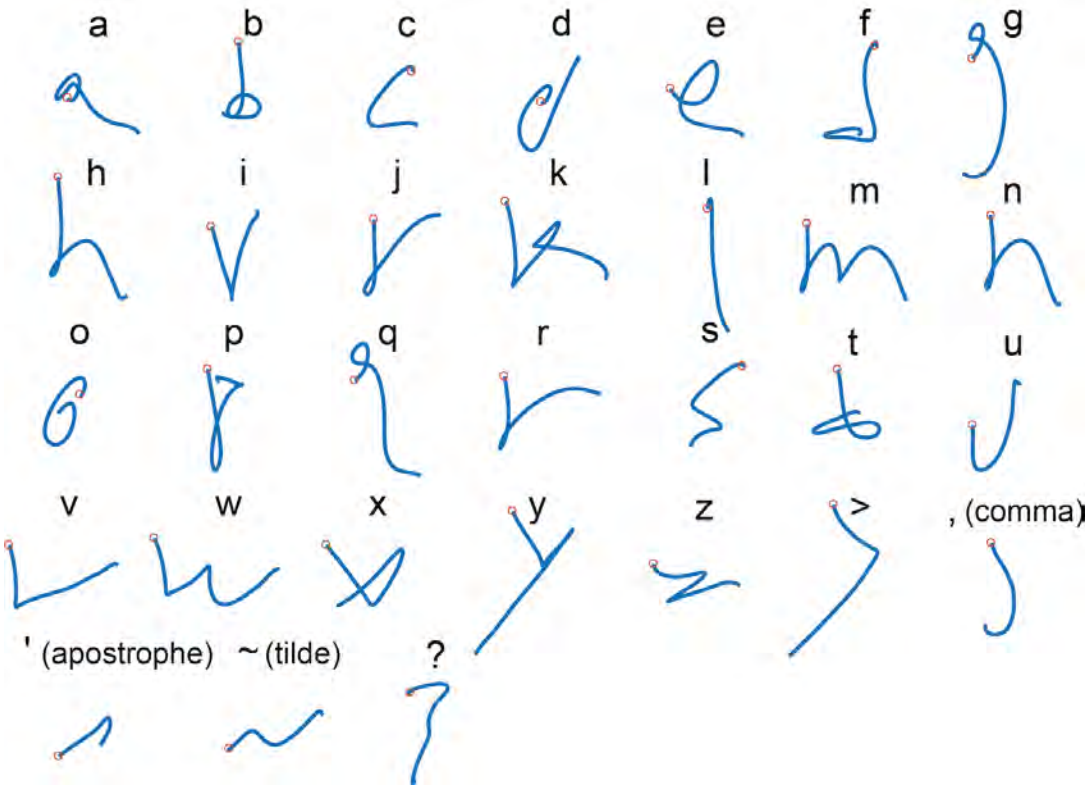
The participant, who was 65 years old at the time of the research, had a spinal cord injury that left him paralyzed from the neck down. Using signals the sensors picked up from individual neurons when the man imagined writing, a machine learning algorithm recognized the patterns his brain produced with

each letter. With this system, the man could copy sentences and answer questions at a rate similar to that of someone his age typing on a smartphone.

This so-called “Brain-to-Text” BCI is so fast because each letter elicits a highly distinctive activity pattern, making it relatively easy for the algorithm to distinguish one from another, Willett says.

A new system

Shenoy’s team envisions using attempted handwriting for text entry as part of a more comprehensive system that also includes point-and-click navigation, much like that used on current smartphones, and even attempted speech decoding. “Having those two or three modes and switching between them is something we naturally do,” he says.



As the participant imagined writing a letter or symbol, sensors implanted in his brain picked up on patterns of electrical activity, which an algorithm interpreted to trace the path of his imaginary pen. Credit: F. Willett et al./*Nature* 2021

Next, Shenoy says, the team intends to work with a participant who cannot speak, such as someone with amyotrophic lateral sclerosis, a degenerative neurological disorder that results in the loss of movement and speech.

The new system could potentially help those suffering from paralysis caused by a number of conditions, Henderson adds. Those include brain stem stroke, which afflicted Jean-Dominique Bauby, the author of the book *The Diving Bell and the Butterfly*. “He was able to write this moving and beautiful

book by selecting characters painstakingly, one at a time, using eye movement," Henderson says. "Imagine what he could have done with Frank's handwriting interface!"

###

Citation

Francis R. Willett, Donald T. Avansino, Leigh R. Hochberg, Jaimie M. Henderson, and Krishna V. Shenoy. "High-performance brain-to-text communication via handwriting." *Nature*. Published online May 12, 2021. doi: 10.1038/s41586-021-03506-2

CAUTION: Investigational Device. Limited by Federal law to investigational use.

Scientist Profiles

Krishna V. Shenoy
Stanford University
Neuroscience Bioengineering

FOR MORE INFORMATION

Meghan Rosen
301-215-8859
rosenm2@hhmi.org

GET HHMI NEWS BY EMAIL

Sign up

Subscribe to RSS

Related Links

The Shenoy Lab

How to "Read" the Brain Signals Underlying Human Speech

Software turns ‘mental handwriting’ into on-screen words, sentences

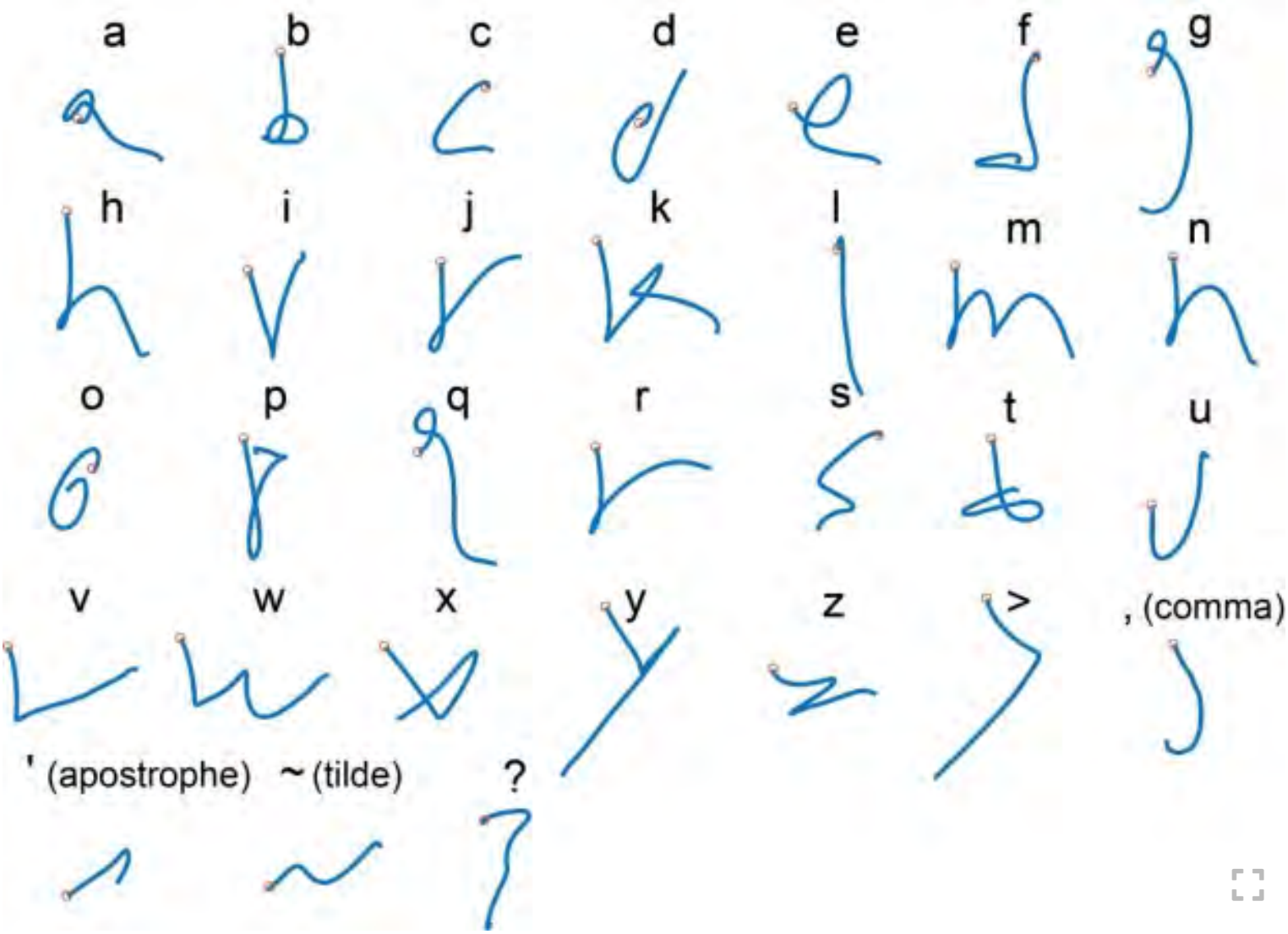
Artificial intelligence, interpreting data from a device placed at the brain’s surface, enables people who are paralyzed or have severely impaired limb movement to communicate by text.

MAY 12
2021

Call it “mindwriting.”

The combination of mental effort and state-of-the-art technology have allowed a man with immobilized limbs to communicate by text at speeds rivaling those achieved by his able-bodied peers texting on a smartphone.

Stanford University investigators have coupled artificial-intelligence software with a device, called a brain-computer interface,



A patient with limb paralysis imagined writing letters of the alphabet. Sensors implanted in his brain picked up the signals, and artificial intelligence algorithms transcribed them onto a computer screen.
Frank Willett

implanted in the brain of a man with full-body paralysis. The software was able to decode information from the BCI to quickly convert the man’s thoughts about handwriting into text on a computer screen.

The man was able to write using this approach more than twice as quickly as he could using a previous method developed by the Stanford researchers, who reported those findings in 2017 in the journal *eLife*.

Hope for those without use of their arms

The new findings, published online May 12 in *Nature*, could spur further advances benefiting hundreds of thousands of Americans, and millions globally, who’ve lost the use of their upper limbs or their ability to speak due to spinal-cord injuries, strokes or amyotrophic lateral sclerosis, also known as Lou Gehrig’s disease, said Jaimie Henderson, MD, professor of neurosurgery.

“This approach allowed a person with paralysis to compose sentences at speeds nearly comparable to those of able-bodied adults of the same age typing on a smartphone,” said Henderson, the John and Jene Blume — Robert and Ruth Halperin Professor." The goal is to restore the ability to communicate by text.”

The participant in the study produced text at a rate of about 18 words per minute. By comparison, able-bodied people of the same age can punch out about 23 words per minute on a smartphone.

The participant, referred to as T5, lost practically all movement below the neck because of a spinal-cord injury in 2007. Nine years later, Henderson placed two brain-computer-interface chips, each the size of a baby aspirin, on the left side of T5’s brain. Each chip has 100 electrodes that pick up signals from neurons firing in the part of the motor cortex — a region of the brain’s outermost surface — that governs hand movement.

Those neural signals are sent via wires to a computer, where artificial-intelligence algorithms decode the signals and surmise T5’s intended hand and finger motion. The algorithms were designed in Stanford’s Neural Prosthetics Translational Lab, co-directed by Henderson and Krishna Shenoy, PhD, professor of electrical engineering and the Hong Seh and Vivian W. M. Lim Professor of Engineering.

Shenoy and Henderson, who have been collaborating on BCIs since 2005, are the senior co-authors of the new study. The lead author is Frank Willett, PhD, a research scientist in the lab and with the Howard Hughes Medical Institute.

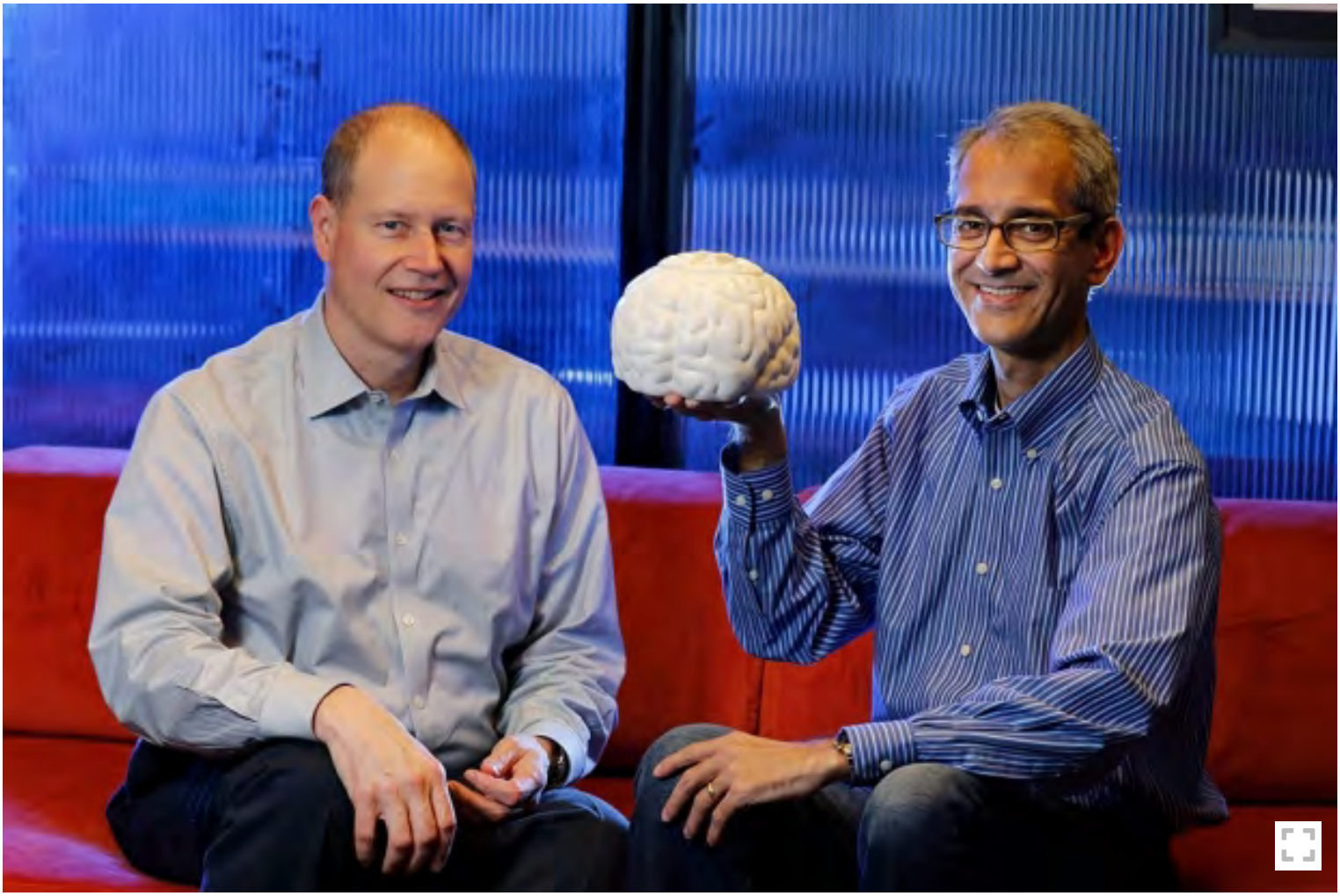
“We’ve learned that the brain retains its ability to prescribe fine movements a full decade after the body has lost its ability to execute those movements,” Willett said. “And we’ve learned that complicated intended motions involving changing speeds and curved trajectories, like handwriting, can be interpreted more easily and more rapidly by the artificial-intelligence algorithms we’re using than can simpler intended motions like moving a cursor in a straight path at a steady speed. Alphabetical letters are different from one another, so they’re easier to tell apart.”

In the 2017 study, three participants with limb paralysis, including T5 — all with BCIs placed in the motor cortex — were asked to concentrate on using an arm and hand to move a cursor from one key to the next on a computer-screen keyboard display, then to focus on clicking on that key.

A mental handwriting speed record

In that study, T5 set what was until now the all-time record: copying displayed sentences at about 40 characters per minute. Another study participant was able to write extemporaneously, selecting whatever words she wanted, at 24.4 characters per minute.

If the paradigm underlying the 2017 study was analogous to typing, the model for the new *Nature* study is analogous to handwriting. T5 concentrated on trying to write individual letters of the alphabet on an imaginary legal pad with an imaginary pen, despite his inability to move his arm or hand. He repeated each letter 10 times, permitting the software to “learn” to recognize the neural signals associated with his effort to write that particular letter.



Jaimie Henderson and Krishna Shenoy, who have been collaborating on brain-computer interfaces since 2005, are the co-senior authors of a study describing the new work.
Paul Sakuma

In further sessions, T5 was instructed to copy sentences the algorithms had never been exposed to. He was eventually able to generate 90 characters, or about 18 words, per minute. Later, asked to give his answers to open-ended questions, which required some pauses for thought, he generated 73.8 characters (close to 15 words, on average) per minute, tripling the previous free-composition record set in the 2017 study.

T5’s sentence-copying error rate was about one mistake in every 18 or 19 attempted characters. His free-composition error rate was about one in every 11 or 12 characters. When the researchers used an after-the-fact autocorrect function — similar to the ones incorporated into our smartphone keyboards — to clean things up, those error rates were markedly lower: below 1% for copying, and just over 2% for freestyle.

These error rates are quite low compared with other BCIs, said Shenoy, who is also a Howard Hughes Medical Institute investigator.

“While handwriting can approach 20 words per minute, we tend to speak around 125 words per minute, and this is another exciting direction that complements handwriting. If combined, these systems could together offer even more options for patients to communicate effectively,” Shenoy said.

The BCI used in the study is limited by law to investigational use and is not yet approved for commercial use.

Stanford University’s Office of Technology Licensing has applied for a patent on intellectual property associated with Willett, Henderson and Shenoy’s work.

Henderson and Shenoy are members of the Wu Tsai Neurosciences Institute at Stanford and of Stanford Bio-X.

Donald Avansino, PhD, a software engineer in the Neural Prosthetics Translational Lab, was a co-author of the study.

The study’s results are the latest chapter of a long-running collaboration between Henderson and Shenoy and a multi-institutional consortium and clinical trial called BrainGate2 (NCT00912041). Study co-author Leigh Hochberg, MD, PhD, a neurologist and neuroscientist at Massachusetts General Hospital, Brown University and the Veterans Affairs Providence Health Care System in Rhode Island, is the sponsor-investigator of BrainGate2.

The study was funded by the Wu Tsai Neurosciences Institute, the Howard Hughes Medical Institute, the U.S. Department of Veterans Affairs, the National Institutes of Health (grants UH2NS095548, R01DC009899, R01DC017844, R01DC014034 and U01NS098968), Larry and Pamela Garlick, Samuel and Betsy Reeves, and the Simons Foundation.

By
BRUCE GOLDMAN
Bruce Goldman is a science writer in the Office of Communications. Email him at goldmanb@stanford.edu.

Shots

TREATMENTS

Man Who Is Paralyzed Communicates By Imagining Handwriting

May 12, 2021 · 12:03 PM ET

JON HAMILTON

A man who is paralyzed was able to type with 95% accuracy by imagining that he was handwriting letters on a sheet of paper, a team reported in the journal *Nature*.

Science Photo Library/Pasieka/Getty Images

An experimental device that turns thoughts into text has allowed a man who was left paralyzed by an accident to construct sentences swiftly on a computer screen.

The man was able to type with 95% accuracy just by imagining he was handwriting letters on a sheet of paper, a team reported Wednesday in the journal *Nature*.

"What we found, surprisingly, is that [he] can type at about 90 characters per minute," says Krishna Shenoy of Stanford University and the Howard Hughes Medical Institute.

The device would be most useful to someone who could neither move nor speak, says Dr. Jaimie Henderson, a neurosurgeon at Stanford and co-director, with Shenoy, of the Stanford Neural Prosthetics Translational Laboratory.

"We can also envision it being used by someone who might have had a spinal cord injury who wants to use email," Henderson says, "or, say, a computer programmer who wants to go back to work."

Both Henderson and Shenoy have a proprietary interest in commercializing the experimental approach used to decode brain signals.

The idea of decoding the brain activity involved in handwriting is "just brilliant," says John Ngai, who directs the National Institutes of Health's BRAIN Initiative, which helped fund the research.

"But it was only on one subject in a laboratory setting," Ngai says. "So at the moment it's a great demonstration of proof of principle."

The man who agreed to test the device is unable to move his arms and legs as the result of a freak accident.

"He was taking out the garbage, slipped, fell and instantly became quadriplegic," Henderson says. "So he's essentially completely paralyzed."

A few years ago, the man agreed to take part in a study of an experimental system called BrainGate2. It allows people who are paralyzed to control computers and other devices using only their thoughts.

The system relies on electrodes surgically implanted near the part of the brain that controls movement. In previous studies, participants had learned to control a computer cursor or robotic arm by imagining they were moving their hands.

This time, Henderson, Shenoy and a team of scientists had the man imagine he was writing individual letters by hand while a computer monitored the electrical activity in his brain.

Eventually, the computer learned to decode the distinct pattern of activity associated with every letter of the alphabet as well as several symbols.

Once that process is complete, Shenoy says, "We can determine if the letter you wrote is an A or a B or a C and then plop that up on the screen and you're able to spell out words and sentences and so forth one letter at a time."

In previous experiments, participants had been able to use their thoughts to "point and click" at letters on a screen. But that approach was much slower than imagined handwriting.

Also, because the new system relies on familiar thoughts, the participant was able to use it almost immediately.

"He was very happy when he was able to write out messages in response to the questions we asked him." Henderson says. "He was pretty excited about this."

The team's success decoding imagined handwriting is just the latest advance in efforts to link computers to the human brain, Ngai says.

"I was introduced to this concept over 10 years ago, and I thought it was quite a bit of science fiction," he says. "Then roughly about five years later it was shown to be not to

be such science fiction after all. So I think we're seeing a progression. It's really quite exciting."

An editorial accompanying the study shares that view.

The handwriting approach "has brought neural interfaces that allow rapid communication much closer to a practical reality," wrote Pavithra Rajeswaran and Amy L. Orsborn of the University of Washington.

brain research paralysis neuroscience

More Stories From NPR