Published in final edited form as:
Brain Comput Interfaces (Abingdon). 2018 ; 5(1): 13–22. doi:10.1080/2326263X.2017.1410418.

Improving P300 Spelling Rate using Language Models and Predictive Spelling

William Speier[a,b], **Corey Arnold**[b], **Nand Chandravadia**[c], **Dustin Roberts**[a], **Shrita Pendekanti**[c], and **Nader Pouratian**[a,c,d,e,*]

[a]Department of Neurosurgery, University of California, Los Angeles, USA

[b]Medical Imaging Informatics Group, University of California, Los Angeles, USA

[c]Neuroscience Interdepartmental Program, University of California, Los Angeles, USA

[d]Department of Bioengineering, University of California, Los Angeles, USA

[e]Brain Research Institute, University of California, Los Angeles, USA

Abstract

The P300 Speller Brain-Computer Interface (BCI) provides a means of communication for those suffering from advanced neuromuscular diseases such as amyotrophic lateral sclerosis (ALS). Recent literature has incorporated language-based modelling, which uses previously chosen characters and the structure of natural language to modify the interface and classifier. Two complementary methods of incorporating language models have previously been independently studied: predictive spelling uses language models to generate suggestions of complete words to allow for the selection of multiple characters simultaneously, and language model-based classifiers have used prior characters to create a prior probability distribution over the characters based on how likely they are to follow. In this study, we propose a combined method which extends a language-based classifier to generate prior probabilities for both individual characters and complete words. In order to gauge the efficiency of this new model, results across 12 healthy subjects were measured. Incorporating predictive spelling increased typing speed using the P300 speller, with an average increase of 15.5% in typing rate across subjects, demonstrating that language models can be effectively utilized to create full word suggestions for predictive spelling. When combining predictive spelling with language model classification, typing speed is significantly improved, resulting in better typing performance.

Keywords

Brain-Computer Interfaces; P300 Speller; Electroencephalography; Language Models; Predictive Spelling

[*]Nader Pouratian MD PhD, NPouratian@mednet.ucla.edu, UCLA Neurosurgery, 300 Stein Plaza, Suite 562, Los Angeles, CA 90095, Phone 310-206-2189, FAX 310-794-1848.

Introduction

Neurodegenerative diseases such as ALS restrict an individual's ability to fully engage with his or her surroundings by interrupting crucial cell signalling processes between the brain and the peripheral nervous system. Brain-computer interface systems including the P300 speller present a promising alternative to traditional communication methods by translating neural signals into text, effectively bypassing the affected pathways [1]. Subjects using the P300 focus on a target character in a grid while stimuli consisting of highlighted rows and columns are presented. When the target character is highlighted, a response signal is evoked, which can be detected to determine the target character. Current challenges to the P300 system include a low signal to noise ratio (SNR), which slows down typing speed, as several stimuli are necessary to achieve an accurate signal reading. Studies have attempted to accelerate typing speed by optimizing different aspects of the system, including grid size [2–4], system parameters [5–7], stimulus presentation methods [3,8], signal processing methods [9–12], and stimulus types [13].

The domain of natural language has been well studied in other fields such as speech recognition and this knowledge can be used to aid in any communication system [14]. By modelling the patterns and structures of natural language, typing speed and accuracy can be improved, and other features such as word completion or automatic error correction can be added [15]. One language-based method that has been shown to significantly improve the speed of BCI systems is predictive spelling (PS), which allows users to type completed words. Similar to methods used in text messaging [16] and augmentative and alternative communication (AAC) devices [17], systems with PS analyse previous character selections to suggest full words to the user. One of the earliest implementations was presented by Ryan et al. [18], who directed P300 output to Quillsoft WordQ2 (version 2.5, Quillsoft, Ltd, Toronto, ON), assistive software which suggested word completions, which could then be selected by typing corresponding numbers from the standard interface. This implementation offered a significant improvement in the number of characters typed per minute in comparison to the standard paradigm, but had lower selection accuracy. Kaufmann et al. [19] implemented a similar approach, which integrated PS into the graphical interface of the P300 speller by replacing numbers with the most common words from a corpus of German newspaper articles. Streamlining the presentation of the suggested selections significantly improved the number of characters selected per minute, but also maintained the accuracy from the non-PS paradigm.

While these PS implementations have been shown to improve performance over the standard system, they have the shortcoming that they give all suggested words the same weight during selection, regardless of their relative likelihood. Another effective implementation of language models in the P300 speller has been to provide prior probabilities for character selections. These models use corpora of text to determine character probabilities based on those previously selected. Early examples used naïve Bayes or hidden Markov models to incorporate n-gram language models, which demonstrated significant improvements in system speed and accuracy [20–25]. More sophisticated language models have been implemented using sampling methods such as particle filtering (PF) to further improve accuracy by giving stronger prior probabilities to target characters and automatically correcting errors [26]. We hypothesize that these methods can be extended to provide probabilities for suggested words by weighting them based on their relative frequencies.

The goal of this study was to extend a previously reported PF method for P300 signal classification to create word suggestions for PS [26]. Using a word-based language model, a probability distribution over possible character and word targets is made by sampling the possible targets in the model. The resulting distribution provides both a set of suggested target words, and a probability distribution over the possible selections that is used as a prior probability. We compared online performance using the modified model with that using the standard PF model in a set of healthy subjects to determine whether incorporating PS yields improvements over using language models for prior probabilities alone.

Materials and Methods

Data Collection

All data was acquired using g.tec amplifiers, active EEG electrodes, and electrode cap (Guger Technologies, Graz, Austria); sampled at 256 Hz; referenced to the left ear; grounded to AF_Z; and filtered using a passband of 0.1 – 60 Hz. Additional artifact detection (e.g., eye blink detection) was not performed and it was left to the classifier to determine whether a signal contained a valid ERP. The electrode set consisted of a previously reported set of 32 electrodes [7]. The subjects for the online study consisted of 12 healthy volunteers with normal or corrected to normal vision between the ages of 20 and 35. The system used a 6 × 6 character grid, famous faces stimuli [13], row and column flashes, and a stimulus onset asynchrony of 125 ms. During sessions with PS enabled, suggested words appeared on the top row of the grid and the numbers 1–6 were removed (Figure 1). Using the standard interface, a 3.5-second gap was included between characters to allow subjects time to find the next character in the sequence. When PS was enabled, this gap was increased to five seconds to allow for the additional task of checking suggested word completions for the target word.

Each experimental session consisted of three training trials followed by two online testing trials, one with and one without PS. Each training trial consisted of copy spelling a preselected 10-character phrase. For the online portion, subjects were instructed to decide on a phrase of their choosing that consisted of approximately 10 words. For each of the online trials, the subject had five minutes to spell as much as they could of their phrase using the PF classifier with and without PS enabled. Counterbalancing was realized by flipping a coin to determine whether PS would be enabled in the first or second online trial. Subjects were instructed not to correct errors and to repeat the phrase if they completed it in under five minutes. If the system incorrectly picked a word completion, subjects were instructed to move to the next word rather than attempting to continue spelling the current word.

BCI2000 was used for data acquisition and online analysis [27]. Statistical analysis was performed using MATLAB (version 8.6.0, MathWorks, Inc, Natick, MA).

Language Model

The model of the English language used in this study is identical to the probabilistic automata model described previously by Speier et al. [26]. This model consists of a directed graph with states for every substring that starts a word in the corpus, starting with a blank root node (Figure 2). Each node has directed edges to nodes that add a single character to the string. Thus, a model of a vocabulary consisting only of the word “THE” would result in four states: the root node representing a blank string, “T,” “TH,” and “THE.” When the word “THAT” is added to the model, it shares the root node and the “T” and “TH” states, and adds two additional states: “THA” and “THAT.” The state “TH” then links to both the states “THE” and “THA.”

States that represent completed words contain links back to the root node to begin a new word. The state “THE,” for instance, links to the root because “THE” is a complete word, but it also is the beginning of other words so it has additional links to other states such as “THEM” or “THEY.” Transition probabilities are determined by the relative frequencies of substrings in the Brown English language corpus [28].

$$p\left(x_t \middle| x_{0:t-1}\right) = \frac{c(x_{m:t})}{\sum_{x_t'} c(x_{m:t-1}, x_t')} = \frac{c(x_{m:t})}{c(x_{m:t-1})} \qquad (1)$$

where m is the index of the last root node in the sequence $x_{0:t-1}$, and $c(x_{m:t})$ is the number of occurrences of words that start with the string $x_{m:t}$ in the corpus. For instance, the probability of typing the letter “E” after “TH” has already been entered is found by dividing the number of occurrences of words that begin with “THE” by the number of times words start with “TH” in the corpus. Similarly, the probability that a word ends and the state transitions back to the root is the ratio of the number of times that word occurs in the corpus over the number of word occurrences starting with that substring.

Classifier

Because it is impractical to compute the probability distribution over all possible strings typed by the user in real time, the probability distribution is estimated using the PF classifier. This classifier estimates the probability distribution over possible outputs by sampling a batch of possible realizations of the model (i.e., a batch of output strings that could have been typed by the user). Each of these realizations is called a particle, which contains a pointer to a node in the model and represents one possible configuration of the model at a given time. Each of these particles moves through the language model independently, based on the model transition probabilities. Low probability realizations are periodically replaced by more likely realizations by resampling the particles based on weights derived from the observed EEG responses. The algorithm estimates the probability distribution of the possible output strings by finding the proportion of the particles that point to each state after they have moved through the model.

In order to determine the probability that the user is attempting to type a given character x_t based on the observed signals, stepwise linear discriminant analysis (SWLDA) is used to select a set of signal features to include in a discriminant function [29]. During training, the algorithm uses ordinary least-squares regression to predict class labels and iteratively adds the most significant features and removes the least significant features until either the target number of features was met or it reached a state where no features were added or removed [10]. The score for flash i for character t, y_t^i, can then be computed as the dot product of the feature weight vector with the features from that trial's signal. It has been shown that scores can be approximated as independent samples from a Gaussian distribution given the target character [19].

$$f(y_t^i \middle| x_t) = \begin{cases} \frac{1}{\sqrt{2\pi\sigma_a^2}}\exp\left(-\frac{1}{2\sigma_a^2}\left(y_t^i - \mu_a\right)^2\right) if\ x_t \in A_t^i \\ \frac{1}{\sqrt{2\pi\sigma_n^2}}\exp\left(-\frac{1}{2\sigma_n^2}\left(y_t^i - \mu_n\right)^2\right) if\ x_t \notin A_t^i \end{cases} \quad (2)$$

where μ_a, σ_a^2, μ_n, and σ_n^2 are the means and variances of the distributions for the attended and non-attended flashes, respectively, and A_t^i is the set of characters highlighted in flash i. The conditional probability of a target at time t given the EEG signal and the previous target characters $x_{0:t-1}$ can then be found:

$$p\left(x_t \middle| y_t, x_{0:t-1}\right) \alpha p(y_t \middle| x_t) p(x_t \middle| x_{0:t-1}) \alpha p(x_t \middle| x_{0:t-1}) \prod_i f\left(y_t^i \middle| x_t\right) \quad (3)$$

where $p(x_n|x_{0:n-1})$ is the prior probability of character x_n given the previously selected characters, determined from the language model. Because the previous target characters are unknown, it is necessary to compute the probability over all possible output strings. This computation is impractical, so the distribution needs to be estimated using sampling methods such as particle filtering.

In particle filtering, a set number of samples (i.e., particles) are created to estimate the distribution over the language model. Each particle j consists of a link to a state in the language model, $x_t^{(t)}$; a string consisting of the particle's state history, $x_{0:t}^{(t)}$; the index of the last time the particle was in the root node, m; and a weight, $w^{(j)}$. When the system begins, a set of P particles is generated and each is associated with the root node with an empty history and a weight equal to $1/P$. At the start of a new character, a sample character $x_t^{(t)}$ is drawn for each particle from the proposal distribution defined by the language model's transition probabilities from the particle's history, $x_{0:t-1}^{(t)}$.

$$x_t^{(j)} \sim p(x_t | x_{0:t-1}^{(j)}) \quad (4)$$

where $p(x_t|x_{0:t-1}^{(j)})$ is provided by the language model as in equation 1. When a particle transitions between states, its pointer changes from the previous state in the model, x_{t-1}, to the new state x_t The history for each particle, $x_{0:t}^{(t)}$, is stored to represent the output character sequence associated with that particle. After each stimulus response, the score for that response, y_t^i, is computed and the probability weight is updated for each of the particles:

$$w_t^{(j)} \alpha p\left(y_t \middle| x_t^{(j)}\right) \alpha \prod_{\mathrm{i}} f\left(y_t^i \middle| x_t\right) \quad (5)$$

where $f\left(y_t^i \middle| x_n^{(j)}\right)$ is computed as in equation 2. The weights are then normalized and the probability of an output string is found by summing the weights of all particles that correspond to that string.

$$p\left(x_{0:t} \middle| y_{1:t}\right) = \sum_k w_t^{(k)} \delta_{x_{0:t}}^{x_{0:t}^{(k)}} \quad (6)$$

where δ is the Kronecker delta. Dynamic classification was implemented by setting a threshold probability, p_{thresh}, to determine when a decision should be made. The program flashes characters until either $\max_{x_t} p\left(x_t \middle| y_{1:t}\right) = \max_{x_t} \Sigma_{x_{0:t-1}} p\left(x_{0:t} \middle| y_{1:t}\right) \geq p_{\text{thresh}}$ or the number of sets of flashes reached the maximum (10). The classifier then selects the string that satisfied $\operatorname{argmax}_{x_{0:t}} p(x_{0:t}|y_{1:t})$. If characters in this output differ from the previous output text, the previous characters are assumed to be errors and are replaced by those in the current string. A new batch of particles, x_{t}^*, are then sampled from the current particles, x_t, based on the weight distribution, $\boldsymbol{w}_t$. Each of the new particles are then assigned an equal weight $w_{\mathrm{t}}^{*(j)}=1/P$. The subject then moves on to the next character and the process then repeats with the new batch of particles. The optimization of p_{thresh} is impractical for online experiments, so a previously reported value of 0.95 was used for all trials [17].

Predictive Spelling

When PS is added to the model, the same classifier and language model are used, but the projection step is modified in order to estimate the probabilities of potential completed words. When particles are being projected, a proportion, ρ, of them continue moving throughout the model until they reach the root node. Figure 3 contains pseudo-code describing the process of particle projection in the PS enabled model. Note that because particles can move multiple steps in one transition, the length of the particle history can now be greater than t, so it is denoted n. Each particle can have different values of n and m; the subscript j is omitted for these values here for simplicity.

After projection, the probability distribution over words is found by summing the weights of particles that have been projected forward to completed words

$$p(x \mid y_{1:t}) = \sum_{\left\{k : x_n^{(k)} = '\_'\right\}} w_t^{(k)} \delta_x^{x_{m:n}^{(k)}} \quad (7)$$

The top K of these words are then added to designated locations in the character grid (Figure 1). EEG responses associated with flashing those cells are applied to the particles that have been projected to those words. Particles that were projected to lower probability words are given zero probability and will be replaced during the next resample phase. In this study, the probability of a complete word selection was set empirically to 0.40 and six word suggestions were presented to the user.

Evaluation

Evaluation of a BCI system must take into account two factors: the ability of the system to achieve the desired result and the amount of time required to reach that result. Because there is a trade-off between speed and accuracy, evaluation in BCI communication literature is traditionally based on the mutual information between the selected character, x, and the target character, z, referred to as the bit rate (BR).

$$BR = \sum_z p(z) \sum_x p(x \mid z) \log \frac{p(x|z)}{p(x)} \quad (8)$$

In the most common metric, information transfer rate (ITR), the probabilities for all characters are assumed to be the same $(p(x) = \frac{1}{N}$ where N is the size of the alphabet (36)) and errors are assumed to be uniform across all possible characters, so

$$p(x \mid z) = \begin{cases} ACC_c & x = z \\ \dfrac{1 - ACC_c}{N - 1} & x \neq z \end{cases} \quad (9)$$

where$ACC_c = \frac{\Sigma_t \delta_{x_t}^{z_t}}{n}$ is the single character accuracy and n is the total number of characters selected. This reduces the bit rate to

$$BR = \log N + ACC_c \log ACC_c + (1 - ACC_c) \log \frac{1 - ACC_c}{N - 1} \quad (10)$$

This is then multiplied by the average number of characters selected per minute (CPM=n/time) to produce the ITR [26].

$$ITR = BR * CPM \quad (11)$$

One problem introduced by including PS is that sentences including erroneous word completions could be a different length from the target. Comparing at the character level no longer works in this case. One solution is to base accuracy on Levenshtein distance (LD) (i.e., the minimum number of insertions, deletions, and replacements required to convert x into z) [30]. We then have $ACC_c = \frac{n - LD(x, z)}{n}$ and the equations above hold.

It has previously been pointed out that ITR overestimates the amount of information conveyed by the system because characters do not occur with equal frequency [31]. Also, the amount of information that ITR assigns to a word is based largely on the word's length. This metric assigns a significantly higher amount of information to incorrect strings that are share characters to the target, regardless of whether they make syntactic sense or possibly confuse the meaning (table 1). An alternative would be to base the metric on word frequency $\left(p(Z') = \frac{c(z')}{c(*)}\right)$. The accuracy can then be computed as the fraction of correct words $\left(ACC_W = \frac{\Sigma_t \delta_{x'_t}^{z'_t}}{n'}\right)$, resulting in a conditional probability of a selection:

$$p(x' \mid z') = \begin{cases} ACC_W & y = z' \\ (1 - ACC_W)\frac{p(x')}{1 - p(z')} & y \neq z' \end{cases} \quad (12)$$

The bit rate then becomes

$$BR' = \sum_{z'} p(z')\left(ACC_W \log \frac{ACC_W}{p(z')} + (1 - ACC_W)\log \frac{1 - ACC_W}{1 - p(z')}\right) \quad (13)$$

The mutual information can then be found by multiplying by the words selected per minute $\left(WPM = \frac{n'}{time}\right)$.

$$MI = BR' * WPM \quad (14)$$

Because the distributions for speeds, accuracies, and bit rates are not normally distributed, significance was tested for all metrics using Wilcoxon signed-rank tests.

Results

EEG response

Subjects demonstrated a negative inflection in their EEG responses at a latency of 200 ms, and a positive inflection at a latency of 300 ms (Figure 4). These responses are consistent with the N200 and P300 responses that have previously been reported when using famous faces stimuli [13]. No significant difference was found between the EEG responses when subjects were focusing on single characters versus the suggested completed words.

Online performance

Using traditional evaluation metrics, all 12 subjects were able to type characters with at least 80% accuracy using each of the algorithms and all but one of the subjects were able to type at least 10 characters per minute (Table 2). When PS was enabled, 6 of the 12 subjects achieved at least 95% accuracy and a typing speed over 12 characters/minute.

Nine of 12 subjects achieved a higher bit rate when using PS than when using the PF method alone. When using the PF algorithm alone, subjects selected and average of 11.16 characters/ minute with 96.79% accuracy, resulting in an average bit rate of 53.89 bits/ minute. When incorporating PS, subjects achieved significant speed improvements, with an average CPM of 12.72 characters/minute (p=0.002) and an average bit rate of 59.39 bits/ minute (p=0.046). PS resulted in a small accuracy decrease that was not statistically significant (p=0.71).

When using word-level metrics, 10 of 12 subjects achieved a higher bit rate when using PS than when using the PF method alone. Using the PF algorithm, subjects typed an average of 2.19 words/minute with 89.86% accuracy, resulting in an average bit rate of 13.79 bits/ minute. Incorporating PS resulted in significant speed improvements, with an average WPM of 2.53 words/minute (p<0.0001) and an average bit rate of 16.54 bits/minute (p=0.0012). When considered on the word level, PS also saw a small accuracy increase that was not statistically significant (p=0.21).

Discussion

Overall, incorporating PS increased typing speed using the P300 speller, with an average increase of 15.5% in typing rate across subjects. The speed increase of 1.6 characters/minute on average was comparable to the previous studies by Ryan et al. (1.5 characters/minute) [18] and Kaufmann et al. (1.6 characters/minute) [19], although from a much higher baseline (11.2 characters/minute compared to 3.76 characters/minute and 2.01 characters/minute, respectively). This increase was primarily a result of the ability to choose multiple characters at once. The actual rate of selections decreased, mainly due to the extra time allotted between characters for checking the suggested words, but the additional characters typed during word completions more than offset this decrease (Table 4). The amount of benefit provided by PS is largely tied to the length of the words the subject wishes to spell and the frequency of the words in the corpus, which influences how many characters the subject must type before it becomes a suggestion. For uncommon words, the PS method was actually detrimental to typing rate as subjects were required to type out most or all characters at a lower speed. In aggregate, however, PS was beneficial as all but one of the subjects saw increased WPM values.

By incorporating PS, word accuracy increased from 89.85% to 92.56%, while character accuracy decreased from 96.79% to 94.80%. While this decrease was not statistically significant, it could have occurred because incorrect word completions can be drastically different from the target word, resulting in several incorrect characters in the same word. Output using PS therefore has fewer incorrect words, but those words that are typed incorrectly often have more errors than when typing without PS. It is possible that incorrect words can contain some additional information about the word the user was attempting, which could mean that words that are close to the target could convey more information than those with multiple errors. However, this information is usually dependent on the target and erroneous words as well as the surrounding context. For instance, erroneously replacing a word with a different part of speech can make the error obvious, allowing the reader to use context to figure out the target. If the error is the same part of speech as the target, however, the new sentence can be grammatically correct, but with different meaning. Future studies can analyse a reader's ability to understand the meaning of typed strings in the presence of errors to determine the true effect on the information conveyed.

The benefit of the PS option is tied directly to the user's ability and preference to use it. Even with PS enabled, the user has the choice to ignore suggestions and instead continue to spell out words one character at a time. If PS is enabled and not used, it likely reduces spelling speed because of the increased pause between characters. It can also reduce accuracy because a fraction of the particles are reserved for selections, so the system is effectively operating on a reduced number of particles and, therefore, a less precise estimation of the probability distribution. For instance, the phrase chosen by subject L contained the word 'WANT,' which after two selections, was included as one of the suggested options. The subject instead spelled out the word using individual characters despite the fact that the correct word remained in the list of completions for each of the last three selections. The inability for this subject to locate suggestions likely contributed to the fact that he had slower typing speed using PS. Increased use could have allowed this subject to become more familiar with the system and therefore take full advantage of the potential improvements PS provides. In another instance, subject J chose a six-word phrase where two of the words were relatively low probability in the model and were never offered as completions. This subject was therefore required to type these words out completely, resulting in a lower typing speed when PS was enabled. A corpus more targeted towards words the specific user is likely to type would make typed words appear as options sooner, thereby improving the performance of a system with PS.

Limitations and future directions

The language model used in the current system does not allow for words that are outside of the vocabulary (OOV) because they did not appear in the training corpus. Previous models have allowed for such words by using character patterns, such as n-grams, rather than requiring full words from the corpus [20,24,32]. However, these methods have been shown to be less effective than the model used in this study [26]. A model that has the capabilities of both of these frameworks can be created by introducing smoothing, which effectively uses the word-based model for words that appeared in the corpus, but then reverts to a character-based model for OOV words. Similar methods have been previously used for smoothing between high dimensional character models to simpler ones [33,34]. Implementing such a method would be advantageous in a realistic setting where subjects are likely to want to use words that are uncommon in general language such as proper nouns.

Because EEG signals are susceptible to various sources of noise, it could be beneficial to add filters specifically designed to remove artifacts. Artifacts that are uncorrelated with the target stimulus (e.g., background noise, wire movement, spurious eye blinks) would likely decrease signal-to-noise, thereby reducing the accuracy of the system. If artifacts are consistent and correlated with the target stimulus (e.g., the subject moves or blinks after every target stimulus), then they may artificially inflate system performance. While we did not observe movements or unusual blinking patterns by subjects during trials, future studies could use monitors such as eye trackers to verify that this was not taking place.

This study was conducted using healthy volunteers who did not have the same constraints as “locked-in” patients, such as restrictions to eye gaze. While the classifier used in this study was previously tested in the ALS population [15], it is unclear whether the added requirement of checking word suggestions will be more difficult and therefore offset the gains seen by typing multiple characters at once. The healthy subjects in this study generally had no problems with the additional cognitive task of scanning through the suggested words, and therefore appreciated the added speed that predictive text afforded. However, it is possible that this additional task will make the system more taxing for ALS patients, which could make it less practical despite the performance increase. Commercial systems based on eye tracking such as the Tobii Dynavox system (Tobii Technology, Inc., Stockholm, Sweden) already incorporate word suggestions, so it is likely that PS will be beneficial in the target population. However, future studies in the ALS population should be conducted to determine how these results in healthy subjects translate to affected population. If predictive text is a hindrance to some subjects, subjects still have the option to ignore the suggestions and type out individual characters, so incorporating predictive text should not ever hinder a subject’s ability to use the system.

Conclusion

Language models used for improving classification speed and accuracy in the P300 speller can be effectively utilized to create full word suggestions for PS. When combining PS with language model classification, typing speed is significantly improved, resulting in better typing performance. Using these methods can make evaluation difficult because the assumptions of traditional metrics are violated. Evaluating on a word level can overcome some of these difficulties to more accurately evaluate P300 performance.

Acknowledgements

This work was supported by the National Institute of Biomedical Imaging and Bioengineering Award Number K23EB014326.

Biography

William Speier received a B.S. in biomedical engineering and applied mathematics and an M.S. in computer science from Johns Hopkins University, and a Ph.D. in biomedical engineering from UCLA. He is currently a postdoctoral scholar in the neurosurgery department at UCLA and is involved in several areas of research, including medical informatics, neural signal analysis, and brain–computer interfaces.

Corey Arnold received a B.S. in biomedical engineering from the University of Wisconsin - Madison and a Ph.D. in Information Science from UCLA. He is currently an Assistant Professor in the UCLA Department of Radiological Sciences and is a member of the UCLA Medical Imaging Informatics group. A central component of his research entails the investigation of probabilistic graphical models of medical data to enable new modes of information retrieval and knowledge discovery.

Nand Chandravadia received a neuroscience B.S. at UCLA and is currently a research assistant in the UCLA neurosurgery department. His research interests include brain-computer interfaces, brain mapping, and neuroadaptive technologies.

Dustin G. Roberts received a B.S. in biochemistry from UCLA in 2014 and is pursuing an M.D. at the UCLA David Geffen School of Medicine with an interest in neurosurgery. Currently, his research interests include brain-computer interfaces, neuroimaging analysis, and neurosurgical interventions for pain relief.

Shrita Pendekanti is an undergraduate student pursuing a B.S. in neuroscience at UCLA and is currently a research assistant in the UCLA neurosurgery department. Her research interests include functional mapping, brain-computer interfaces, morphological studies, and meta-analyses of brain function.

Nader Pouratian received a B.S. and Ph.D. in neuroscience and an M.D. from UCLA. He is an Associate Professor of neurosurgery, neuroscience, and bioengineering at UCLA. His research focuses on multimodality brain mapping and signal processing including both invasive and noninvasive human neurophysiology to improve neurosurgical safety and efficacy and to develop novel and improved methods to restore communication and function to patients with neurological impairments.

References

[1]. Farwell LA and Donchin E 1988 Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials Electroencephalogr. Clin. Neurophysiol 70 510–23

[2]. Jin J, Allison BZ, Sellers EW, Brunner C, Horki P, Wang X and Neuper C 2011 Optimized stimulus presentation patterns for an event-related potential EEG-based brain-computer interface. Med. Biol. Eng. Comput 49 181–91 [PubMed: 20890671]

[3]. Townsend G, LaPallo BK, Boulay CB, Krusienski DJ, Frye GE, Hauser CK, Schwartz NE, Vaughan TM, Wolpaw JR and Sellers EW 2010 A novel P300-based brain-computer interface stimulus presentation paradigm: Moving beyond rows and columns Clin. Neurophysiol 121 1109–20 [PubMed: 20347387]

[4]. Sellers EWE, Krusienski DJDJ, McFarland DJ, Vaughan TM and Wolpaw JR 2006 A P300 event-related potential brain-computer interface (BCI): The effects of matrix size and inter stimulus interval on performance Biol. Psychol 73 242–52 [PubMed: 16860920]

[5]. McFarland DJ, Sarnacki WA and Wolpaw JR 2003 Brain-computer interface (BCI) operation: optimizing information transfer rates Biol. Psychol 63 237–51 [PubMed: 12853169]

[6]. McFarland DJ, Sarnacki WA, Townsend G, Vaughan T and Wolpaw JR 2011 The P300-based brain-computer interface (BCI): Effects of stimulus rate Clin. Neurophysiol. 122 731–7

[7]. Lu J, Speier W, Hu X and Pouratian N 2013 The effects of stimulus timing features on P300 speller performance Clin. Neurophysiol 124 306–14 [PubMed: 22939456]

[8]. Jin J, Horki P, Brunner C, Wang X, Neuper C and Pfurtscheller G 2010 A new P300 stimulus presentation pattern for EEG-based spelling systems Biomed. Tech 55 203–10

[9]. Kaper M, Meinicke P, Grossekathoefer U, Lingner T and Ritter H 2004 BCI Competition 2003 - Data Set IIb: Support Vector Machines for the P300 Speller Paradigm IEEE Trans. Biomed. Eng 50 1073–6

[10]. Krusienski DJ, Sellers EW, Cabestaing F, Bayoudh S, McFarland DJ, Vaughan TM and Wolpaw JR 2006 A comparison of classification techniques for the P300 Speller J. Neural Eng. 3 299–305 [PubMed: 17124334]

[11]. Xu N, Gao X, Hong B, Miao X, Gao S and Yang F 2004 BCI Competition 2003 - Data Set IIb: Enhancing P300 Wave Detection Using ICA-Based Subspace Projections for BCI Applications IEEE Trans. Biomed. Eng 51 1067–72

[12]. Serby H, Yom-Tov E and Inbar GF 2005 An improved P300-based brain-computer interface IEEE Trans. Neural Syst. Rehabil. Eng 13 89–98

[13]. Kaufmann T, Schulz SM, Grünzinger C, Kübler A 2011 Flashing characters with famous faces improves ERP-based brain–computer interface performance. J Neural Eng. 8 56016

[14]. Jelinek F 1998 Statistical methods for speech recognition (MIT Press)

[15]. Speier W, Chandravadia N, Roberts D, Pendekanti S and Pouratian N 2016 Online BCI Typing using Language Model Classifiers by ALS Patients in their Homes Brain-Computer Interfaces 4 114–121 [PubMed: 29051907]

[16]. Dunlop MD and Crossan A 2000 Predictive text entry methods for mobile phones Pers. Technol 4 134–43

[17]. Darragh JJ, Witten IH and James ML 1990 The reactive keyboard: A predictive typing aid Computer (Long. Beach. Calif). 23 41–9

[18]. Ryan DB, Frye GE, Townsend G, Berry DR, Mesa GS, Gates NA and Sellers EW 2010 Predictive spelling with a P300-based brain–computer interface: increasing the rate of communication Intl. J. Human–Computer Interact 27 69–84

[19]. Kaufmann T, Völker S, Gunesch L and Kübler A 2012 Spelling is just a click away–a user-centered brain–computer interface including auto-calibration and predictive text entry Front. Neurosci 6 72

[20]. Speier W, Arnold C, Lu J, Taira RK and Pouratian N 2011 Natural language processing with dynamic classification improves P300 speller accuracy and bit rate J. Neural Eng. 9 16004

[21]. Park J and Kim K-E 2012 A POMDP approach to optimizing P300 speller BCI paradigm Neural Syst. Rehabil. Eng. IEEE Trans. 20 584–94

[22]. Kindermans P-J, Verschore H, Verstraeten D and Schrauwen B 2012 A P300 BCI for the masses: Prior information enables instant unsupervised spelling Advances in Neural Information Processing Systems pp 710–8

[23]. Speier W, Knall J and Pouratian N 2013 Unsupervised training of brain-computer interface systems using expectation maximization Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on (IEEE) pp 707–10

[24]. Speier W, Arnold C, Lu J, Deshpande A and Pouratian N 2014 Integrating language information with a hidden markov model to improve communication rate in the P300 speller IEEE Trans. Neural Syst. Rehabil. Eng. 22 678–84

[25]. Oken BS, Orhan U, Roark B, Erdogmus D, Fowler A, Mooney A, Peters B, Miller M and Fried-Oken MB 2014 Brain–Computer Interface With Language Model–Electroencephalography Fusion for Locked-In Syndrome Neurorehabil. Neural Repair 28 387–94

[26]. Speier W, Arnold C W, Deshpande A, Knall J and Pouratian N 2015 Incorporating advanced language models into the P300 speller using particle filtering J. Neural Eng. 12 46018

[27]. Schalk G, McFarland DJ, Hinterberger T, Birbaumer N and Wolpaw JR 2004 BCI2000: a general-purpose brain-computer interface (BCI) system IEEE Trans. Biomed. Eng. 51 1034–43

[28]. Francis WN and Kucera H 1979 Brown Corpus Manual (Providence, RI: Dept of Linguistics, Brown University)

[29]. Draper NR and Smith H 1981 Applied Regression Analysis (Wiley)

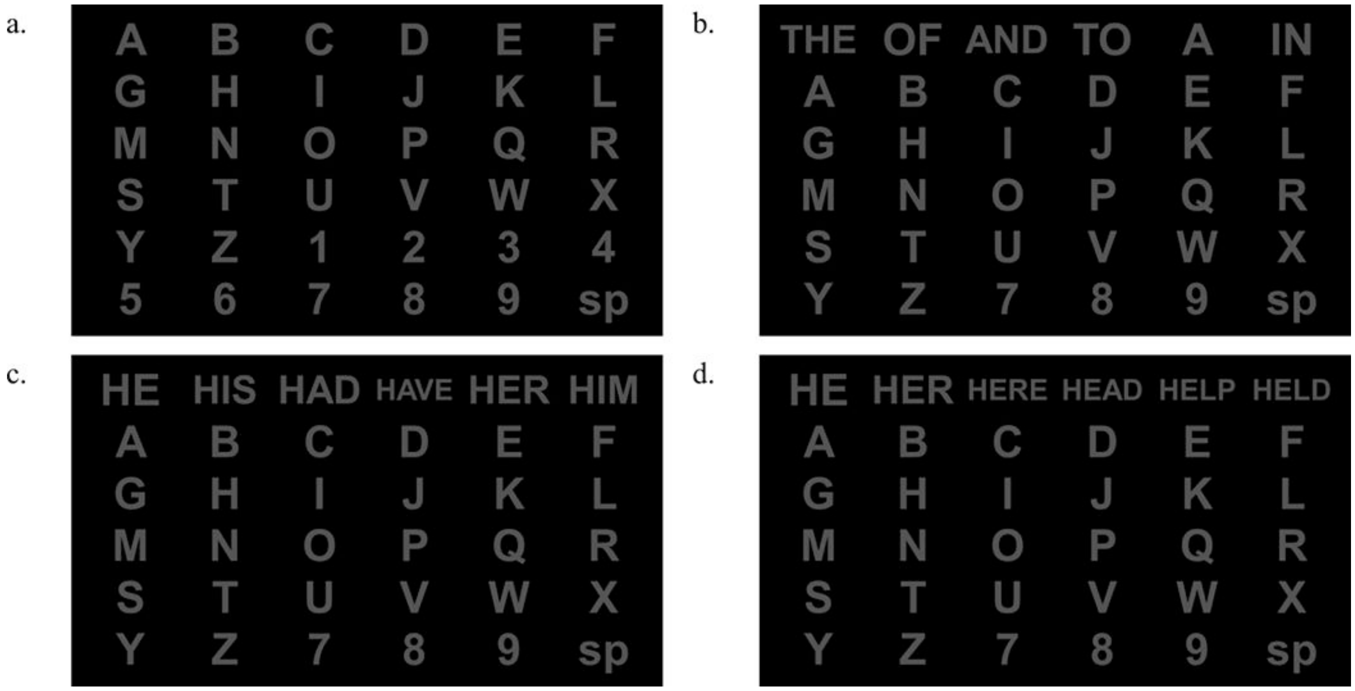
[30]. Levenshtein VI. 1966; Binary codes capable of correcting deletions, insertions and reversals. Soviet physics doklady. 10:707.

[31]. Speier W, Arnold C and Pouratian N 2013 Evaluating True BCI Communication Rate through Mutual Information and Language Models. PLoS One 8 e78432 [PubMed: 24167623]

[32]. Park J and Kim K-E 2012 A POMDP Approach to Optimizing P300 Speller BCI Paradigm IEEE Trans. Neural Syst. Rehabil. Eng. 20 584–94

[33]. Orhan U, Hild KE, Erdogmus D, Roark B, Oken B and Fried-Oken M 2012 RSVP keyboard: An EEG based typing interface Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on (IEEE) pp 645–8

[34]. Kindermans P-J, Verschore H and Schrauwen B 2013 A Unified Probabilistic Approach to Improve Spelling in an Event-Related Potential-Based Brain–Computer Interface Biomed. Eng. IEEE Trans. 60 2696–705



a.

A	B	C	D	E	F
G	H	I	J	K	L
M	N	O	P	Q	R
S	T	U	V	W	X
Y	Z	1	2	3	4
5	6	7	8	9	sp

b.

THE	OF	AND	TO	A	IN
A	B	C	D	E	F
G	H	I	J	K	L
M	N	O	P	Q	R
S	T	U	V	W	X
Y	Z	7	8	9	sp

c.

HE	HIS	HAD	HAVE	HER	HIM
A	B	C	D	E	F
G	H	I	J	K	L
M	N	O	P	Q	R
S	T	U	V	W	X
Y	Z	7	8	9	sp

d.

HE	HER	HERE	HEAD	HELP	HELD
A	B	C	D	E	F
G	H	I	J	K	L
M	N	O	P	Q	R
S	T	U	V	W	X
Y	Z	7	8	9	sp

Figure 1.

Images of the character grids used in standard (a) and predictive spelling (PS) (b-d) trials. In PS trials, the six most likely words are presented in the top row of the grid given the previously typed characters. Three examples are shown with no entered text (b), after entering the letter 'H' (c), and after entering the string "HE" (d).

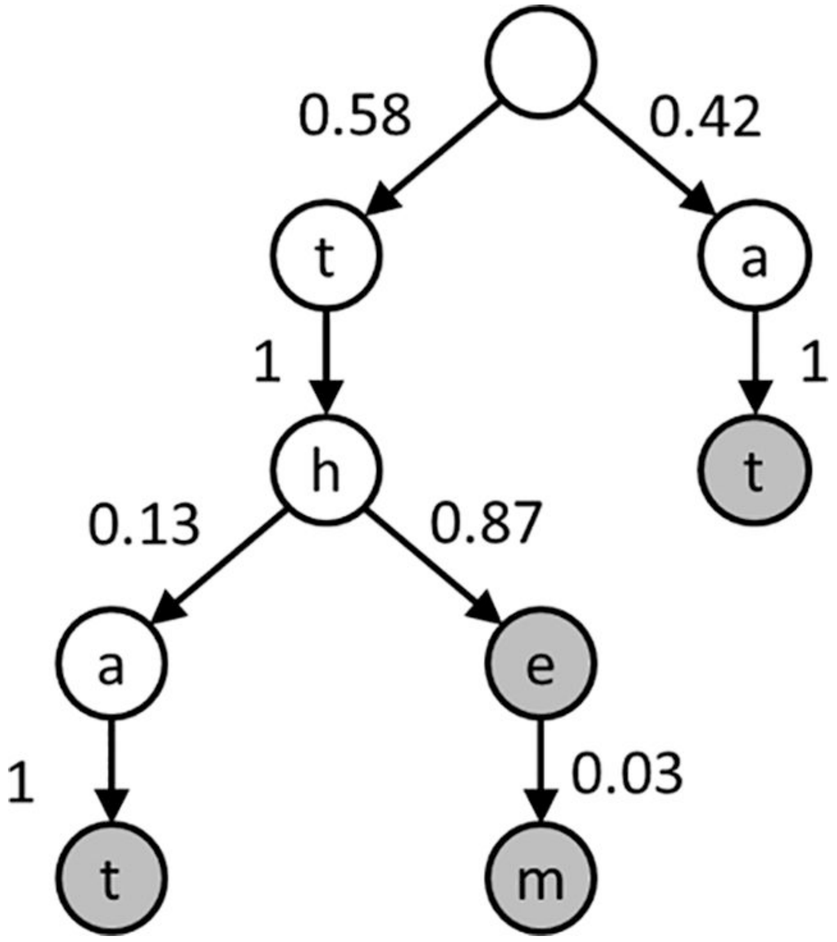
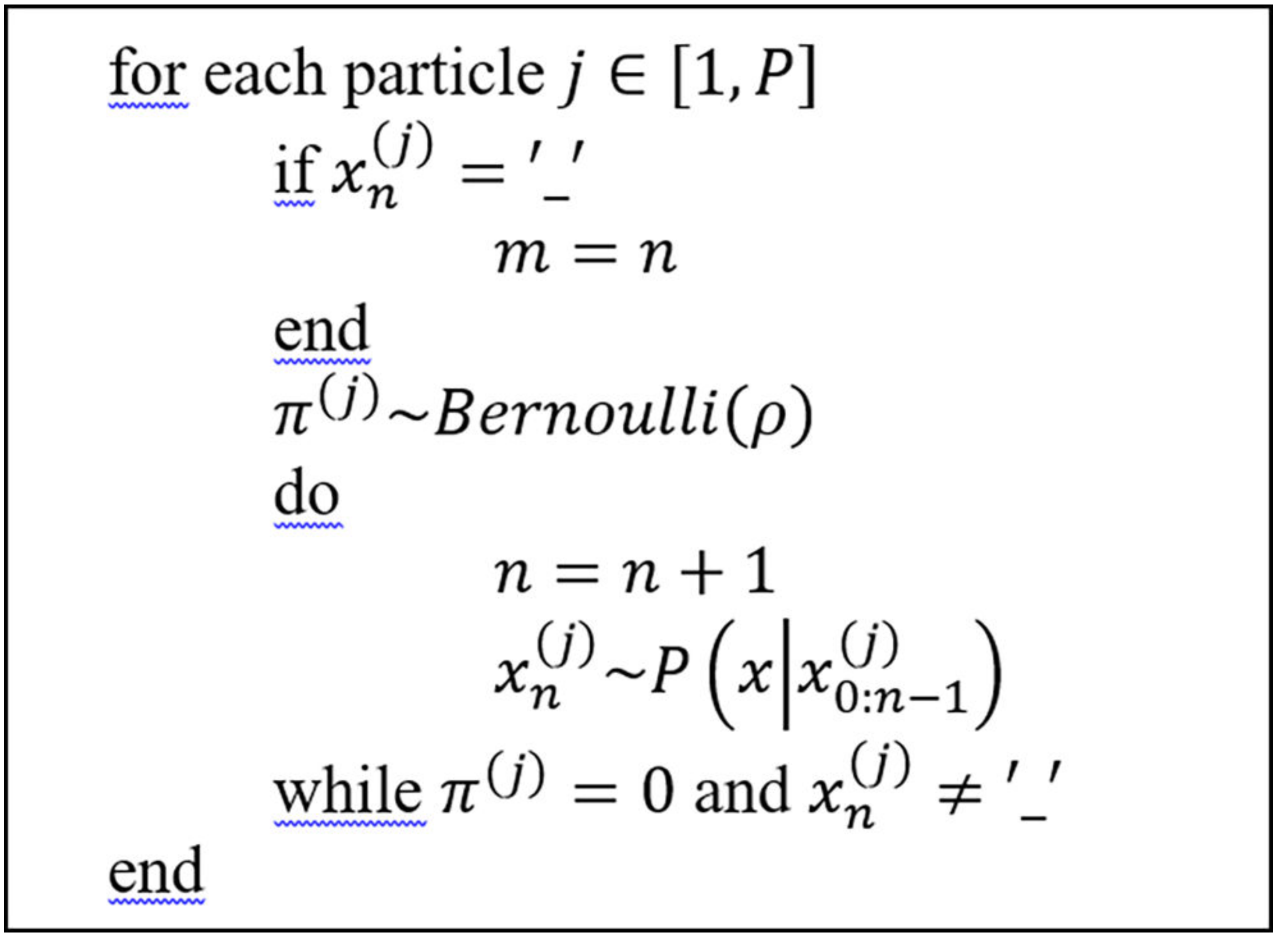


Figure 2.
Example model of a vocabulary consisting of the words “AT,” “THE,” “THEM,” and “THAT.” Shaded states represent complete words that have links back to the root node.



```
for each particle $j \in [1, P]$
    if $x_n^{(j)} = '\_'$
        $m = n$
    end
    $\pi^{(j)} \sim Bernoulli(\rho)$
    do
        $n = n + 1$
        $x_n^{(j)} \sim P\left(x \middle| x_{0:n-1}^{(j)}\right)$
    while $\pi^{(j)} = 0$ and $x_n^{(j)} \neq '\_'$
end
```

Figure 3.
Pseudo-code for projection of particles.

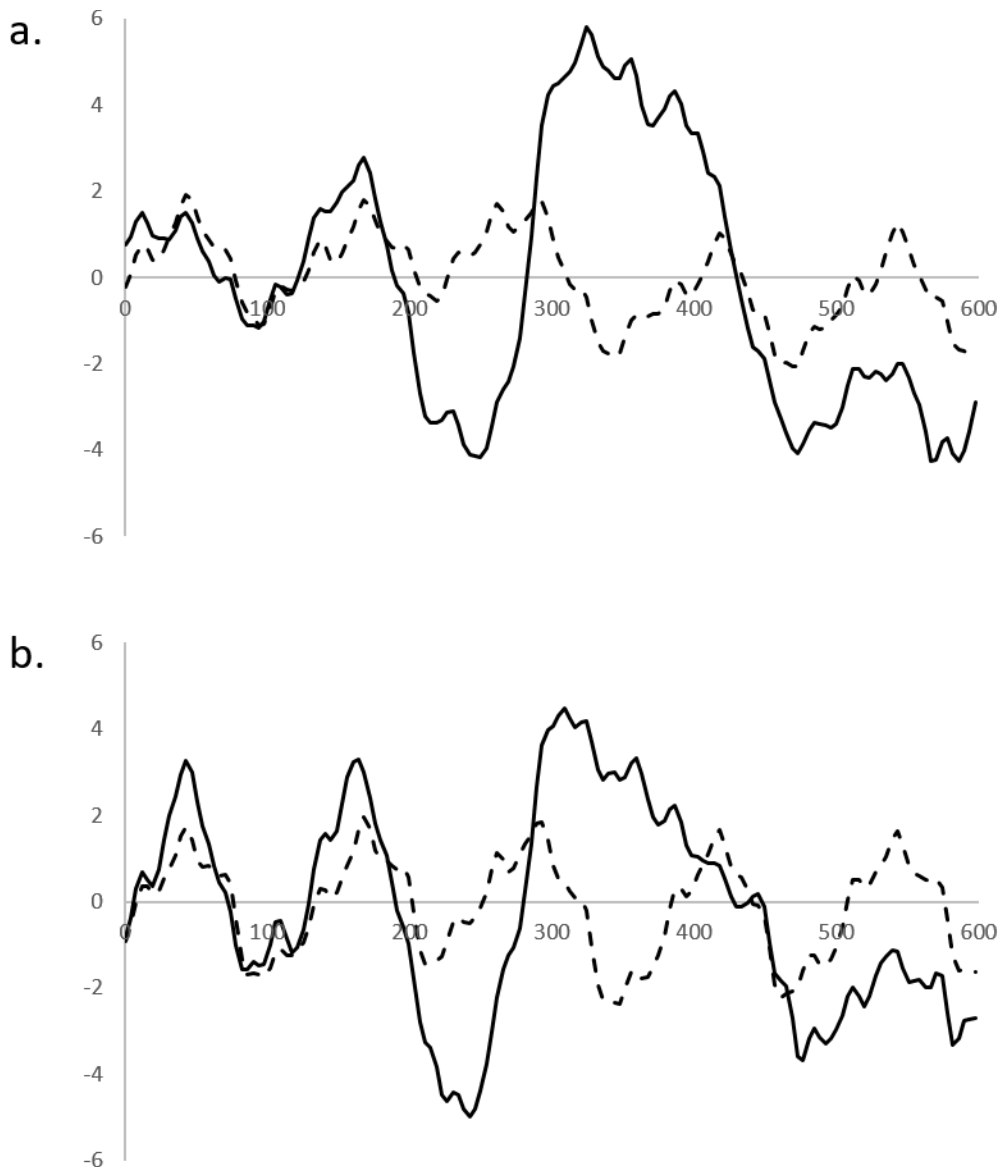


Figure 4.

Average stimulus response for subject B for attended (solid) and non-attended (dashed) stimuli in online trials when attending on a signal character (a) or a completed word (b). Signals are averaged across four channels: CP_Z, PO_Z, PO_7, and PO_8

Table 1.

Bit rate values using the information transfer rate (ITR) and mutual information (MI) methods as well as word and character accuracies (ACC_W and ACC_C, respectively) for example output strings resulting from attempts to type the string "HELLO_WORLD_." The numbers marked with an asterisk are obtained by using Levenshtein distance to compute accuracy. Using the new method, the information rate for incorrect words is the same regardless of word length or similarity to the target.

Typed text	**ACC_C**	**ITR**	**ACC_W**	**MI**
HELLO WORLD_ (target)	100	43.00	100	14.66
HELLO WOULD_	91.7	41.43	50	5.93
HELLO WHIRL_	66.7	27.63	50	5.93
HELLO PLANET_	50*	24.95*	50	5.93

Table 2.

Selection rates (CPM), accuracies (ACCC), and information transfer rates (ITR) for each subject in online spelling using the particle filter (PF) classifier with and without predictive spelling (PS) enabled.

	CPM (characters/minute)		**ACC$_C$ (%)**		**ITR (bits/minute)**	
	PF	**PF-PS**	**PF**	**PF-PS**	**PF**	**PF-PS**
A	11.68	15.07	97.92	100.00	57.45	77.89
B	11.41	14.07	100.00	100.00	58.99	72.72
C	10.34	13.35	98.04	87.69	50.98	53.40
D	11.78	14.27	89.66	96.77	49.02	68.48
E	10.20	11.26	94.00	92.86	46.26	49.90
F	11.30	13.51	98.21	82.09	55.91	48.27
G	10.45	12.07	96.15	100.00	49.49	62.38
H	12.28	14.23	100.00	98.57	63.51	71.01
I	12.23	12.25	96.72	85.51	58.62	46.90
J	9.84	9.56	100.00	100.00	50.86	49.41
K	11.42	12.64	100.00	100.00	59.02	65.34
L	10.96	10.35	90.74	94.12	46.57	47.04
Average	11.16	12.72	96.79	94.80	53.89	59.39

Table 3.

Word level metrics for online trials consisting of the number of words typed per minute (WPM), the percentage of words typed correctly (ACCW), and the mutual information (MI) between the target and typed sentences when using the particle filter (PF) classifier with and without predictive spelling (PS) enabled.

	WPM (words/minute)		**ACC$_W$ (%)**		**MI (bits/minute)**	
	PF	**PF-PS**	**PF**	**PF-PS**	**PF**	**PF-PS**
A	1.87	2.25	83.33	100.00	10.57	16.49
B	2.24	2.78	100.00	100.00	16.43	20.39
C	2.43	3.05	91.67	87.50	15.65	18.40
D	2.37	2.79	75.00	92.86	11.71	18.24
E	1.84	2.41	88.89	83.33	11.33	13.65
F	2.37	2.70	83.33	78.57	13.44	14.14
G	2.21	2.53	81.82	100.00	12.21	18.51
H	2.29	2.65	100.00	92.86	16.80	17.37
I	2.17	2.42	90.91	83.33	13.80	13.69
J	1.90	1.88	100.00	100.00	13.91	13.78
K	2.14	2.37	100.00	100.00	15.69	17.37
L	2.45	2.52	83.33	92.31	13.89	16.38
Average	2.19	2.53	89.86	92.56	13.79	16.54

Table 4.

Example online output for typing with and without predictive spelling (PS). Each row is the result of subject H attempting to spell “WISE MEN SAY FORGIVENESS IS DIVINE BUT NEVER PAY FULL PRICE FOR LATE PIZZA” for five minutes. Bold characters are errors and underlined characters are those selected using PS.

Method	Output
Target	WISE MEN SAY FORGIVENESS IS DIVINE BUT NEVER PAY FULL PRICE FOR LATE PIZZA
PF	WISE MEN SAY FORGIVENESS IS DIVINE BUT NEVER PAY FULL PRICE F
PF-PS	WISE MEN SAY FORGIVENESS IS DIVINE BUT NEVER PAY FULL PRICE F**A**R LATE P