Proceedings of the 6th International Brain-Computer Interface Conference 2014 DOI:10.3217/978-3-85125-378-8-80

Extending Language Modeling to Improve Dynamic Data Collection in ERP-based Spellers

B. O. Mainsah, L. M. Collins, K. D. Morton and C. S. Throckmorton

Department of Electrical and Computer Engineering, Duke University, Durham, NC, U.S.A.
leslie.collins@duke.edu

Abstract

In this study, we extend the bigram language model to a higher order model in our dynamic stopping algorithm for the ERP-based P300 speller, with additional consideration to minimize erroneous character revisions. Prefix alternatives are generated to initialize the language model based on the likelihoods of being the target character. On-line results indicate there is potential to improve ERP speller performance with our proposed method, as statistically significant improvements were observed in participant communication rates.

1 Introduction

Language models can optimize ERP-based BCI speller performance by incorporating the probabilistic information about how letters are ordered in words when making character selection decisions [1–3]. We developed a dynamic stopping algorithm for the ERP-based P300 speller that uses a Bayesian approach to determine the amount of data collection based on a probabilistic level of confidence that a character is the target. The dynamic stopping algorithm with uniform initialization priors resulted in significant increase in bit rate from static data collection [4], with additional improvements observed with the inclusion of a bigram language model in the character initialization process [5]. In this study, we expand to a higher order model that uses all of the user's spelling history, not just the previous character selection, with the hypothesis that it will improve the predictive capacity of the algorithm due to an increasing number of selected characters reducing the set of possible intended characters. However, the utility of a higher order model is dependent on the number of preceding characters that are selected correctly as these are used to obtain the correct target word prefix for initializing subsequent character probabilities. We consider a solution for misspellings by using information about character likelihoods post-data collection [6] to weight possible prefixes.

2 Methods

Participants ($n = 20$) were recruited from the student and work population at Duke University, who gave informed consent prior to participating in the study. Participants performed word copy-spelling tasks online with a 9×8 matrix speller grid using the Bayesian dynamic stopping algorithm (detailed in [5]) with different language models, with order counter-balanced.

Prior to the Bayesian update process, the language model is used to initialize character probabilities given a sequence of previously selected characters, $\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1} = a_{T-n+1}, ..., a_{T-1}$:

$$P(C_{i,T} = C_T^*) = \alpha P(C_{i,T}|\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1}) \left(1 - \sum_{NAC} \frac{1}{N}\right) + (1-\alpha)\frac{1}{N} \tag{1}$$

where $P(C_{i,T} = C_T^*)$ is the initialization probability of character C_i being the T^{th} character in the target word, C_T^*; $P(C_{i,T}|\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1})$ is the probability that the next character is C_i

 DOI:10.3217/978-3-85125-378-8-80

given the previously selected characters $\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1}$, which is based on the order of the language model; α denotes the weight of the language model; $1-\alpha$ denotes the weight of a uniform distribution, which is an error factor to account for possible misspellings; $\sum_{NAC} \frac{1}{N}$ is the sum of the non-alphabetic character (NAC) probabilities, which is subtracted from 1 to normalize the probabilities. The language model probabilities were derived from a corpus compiled by Norvig [7].

2.1 Bigram model

The conditional probability, $P(C_{i,T}|\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1})$, depends on the previously selected character, $\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1} = a_{T-1}$ [5].

2.2 n-gram model with dictionary-assisted prefix search (DAPS)

In the n-gram model, the conditional probability, $P(C_{i,T}|\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1})$, depends on all the previously spelled characters, $\boldsymbol{\mathcal{A}}_{T-n+1}^{T-1} = a_1, ..., a_{T-1}$. However, when using the n-gram model, there is the possibility of an erroneously selected character generating an invalid prefix (e.g. **VIS2** for the word **VISUAL**), or an incorrect valid prefix which can lead to the wrong initialization probabilities (e.g. **ANC** for the word **INCOME**), unless the erroneous character is revised. ERP-based P300 classifier confidences post-data collection can provide some information about the likelihood of being the target character [6]. We denote the P300 classifier confidences with a $\boldsymbol{Q}^T = [Q_1, Q_2, ..., Q_T]$ matrix, where $Q_t = [q_t^1, ..., q_t^N]^\intercal$ is a column vector of P300 classifier confidences post-data collection for all N grid characters for the t^{th} selected character. An example $\boldsymbol{Q}$ matrix is shown in Figure 1. The user intended to spell the word **BEHIND** but the simulation yielded the word **BLGIND**. While the target character may not have the highest probability post-data collection, one of the next most probable characters usually is the target.

The $\boldsymbol{Q}$ matrix can thus be used to select the k most likely prefixes from a dictionary to calculate initialization probabilities prior to spelling the T^{th} character. The set $\boldsymbol{\mathcal{D}}_T^k$ consists of valid prefixes in the dictionary with the top k values of the product of their character likelihoods $\left(\prod_1^{T-1} q_t^{l(D_t^j)}\right)$, and the prefixes are retained from one character to the next to generate the next k most likely prefixes. The conditional probability in the initialization step to select the T^{th} character thus also depends on the P300 classifier confidences via the $\boldsymbol{Q}^{T-1}$ matrix:

$$w(D^j) = \frac{\prod_1^{T-1} q_t^{l(D_t^j)}}{\sum_{D^j \in \boldsymbol{\mathcal{D}}_T^k} \left(\prod_1^{T-1} q_t^{l(D_t^j)}\right)} \tag{2}$$

$$P(C_{i,T}|\boldsymbol{\mathcal{A}}_1^{T-1}, \boldsymbol{Q}^{T-1}) = \sum_{D^j \in \boldsymbol{\mathcal{D}}_T^k} w(D^j) P(C_{i,T}|D^j) \tag{3}$$

where $w(D^j)$ is the weight of the prefix D^j; $l(D_t^j)$ is the label for the t^{th} character in prefix D^j; $P(C_{i,T}|D^j)$ is the initialization probability that the T^{th} character is C_i, given prefix D^j.

3 Results

Figure 2 shows each participant's selection accuracy. For most participants, the accuracy from the bigram to n-gram model was similar or noticeably improved, ($p < 0.09$). The $\boldsymbol{Q}$ matrix tends to be sparse, meaning that only a few characters are considered likely to be the target.

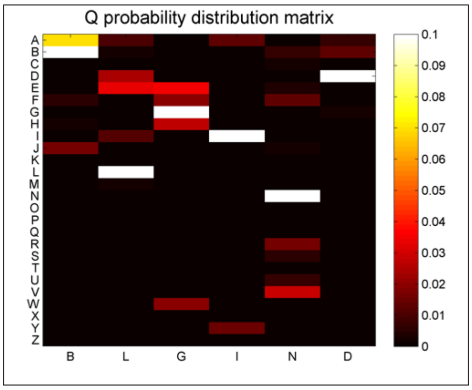

Figure 1: Example $\boldsymbol{Q}$ matrix post-data collection for the word **BEHIND**. The x-axis labels show the characters selected by the ERP-based P300 speller simulation, **BLGIND**, with the corresponding probabilities of alphabet characters at each character position, Q_t. Probability values are clipped for visualization purposes ($P_{max} \geq 0.9$)

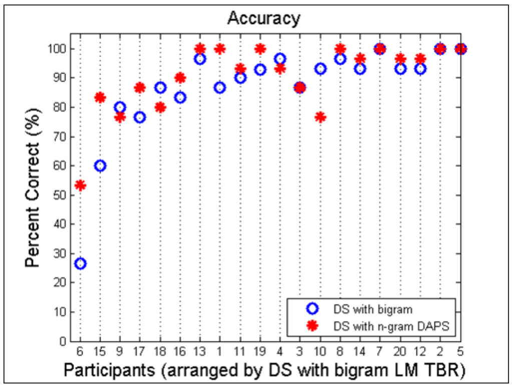

Figure 2: Participant performance comparison of online dynamic stopping between different language models showing accuracy of character selections

These characters typically have at some point been in the same flash group as the target, and this is often the source of most character selection errors. We hypothesize that the strong priors introduced by a higher order language model give the target character an added advantage to prevent an erroneous character selection, thereby sometimes improving accuracy.

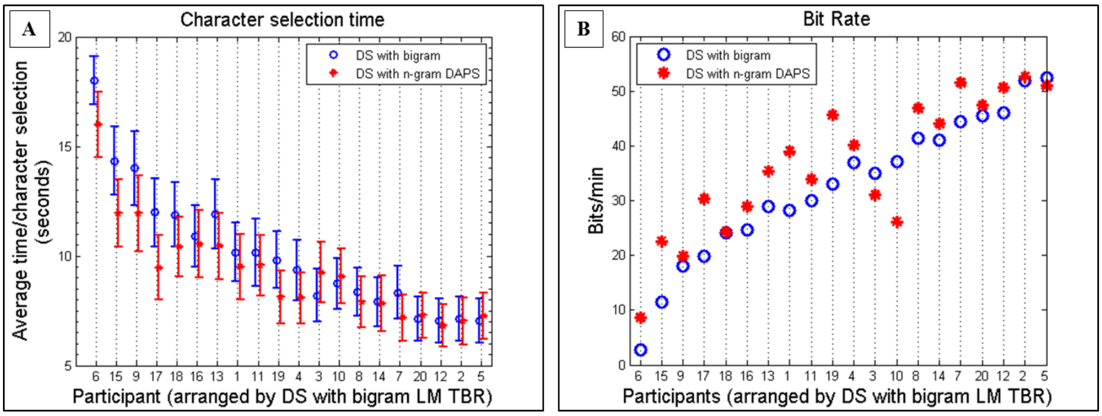

Figure 3: Participant performance comparison between on-line dynamic stopping with different language models. (***A***) Average character selection time, with standard error bars, (***B***) Bit Rate.

.

Figure 3A shows the average amount of time per character selection for each participant. A significant decrease was observed in character selection time with the n-gram DAPS model ($p < 0.002$). In Mainsah *et al.* [5], off-line analysis of participant EEG data revealed that the rate of convergence to the threshold probability in dynamic data collection increased with the inclusion of the bigram language model. We hypothesize that the stronger priors introduced

DOI:10.3217/978-3-85125-378-8-80

by the higher order model further causes the character probabilities to converge faster.

The accuracy and average character selection time were used to calculate bit rate, including the time pauses between character selections [8]. Figure 3B shows participant bit rates with both algorithms. Due to similar accuracy levels and a significant reduction in character selection time, most participants observed significant improvement in their performance ($p < 0.007$), with on average 26% increase in bit rate. Performance improvements with the n-gram model are consistent with off-line analysis performed on EEG data from [5].

4 Discussion

The relatively slow communication rates of ERP-based BCI speller systems can be improved by exploiting the predictability of language. However, sometimes the manner of integration of language information in the ERP speller can lead to a decrease in performance due to increased task difficulty e.g. selecting from a drop-down menu in a predictive speller as in [2]. Our online results indicate there is potential to improve performance with a higher order language model in dynamic data collection, with additional consideration to minimize erroneous character revisions when used in combination with a dictionary. Further development includes adapting the algorithm for sentence spelling tasks, where word-space boundaries are important. There is the potential to further enhance performance using natural language processing tools such as word prediction and/or dictionary-based spelling correction. For example, the algorithm can be adapted to include likely word alternatives generated from the prefixes which can be displayed directly in the speller matrix [3], as this has been shown to not negatively affect performance.

Acknowledgments

This research was supported by NIH/NIDCD grant number R33 DC010470-03.

References

[1] U. Orhan, D. Erdogmus, B. Roark, B. Oken, and M. Fried-Oken. Offline analysis of context contribution to erp-based typing bci performance. *Journal of Neural Engineering*, 10(6), 2013.

[2] D. Ryan, G. Frye, G. Townsend, D. Berry, G. Mesa, N. Gates, and E. Sellers. Predictive spelling with a p300-based brain-computer interface: Increasing the rate of communication. *Int J Hum Comput Interact*, 27(1):69–84, 2011.

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

[4] C. Throckmorton, K. Colwell, D. Ryan, E. Sellers, and L. Collins. Bayesian approach to dynamically controlling data collection in p300 spellers. *IEEE Trans Neural Syst Rehabil Eng*, 21(3):508–17, 2013.

[5] B. Mainsah, K. Colwell, L. Collins, and C. Throckmorton. Utilizing a language model to improve online dynamic data collection in p300 spellers. *IEEE Trans Neural Syst Rehabil Eng*, 22(4), 2014.

[6] B. Mainsah, K. Morton, L. Collins, and C. Throckmorton. Achieving spelling correction in p300 spellers without error-related potentials. *Society for Neuroscience Conference*, 2013.

[7] Peter Norvig. How to write a spelling corrector. `http://norvig.com/spell-correct.html`, 2007.

[8] D.J. McFarland, W.A. Sarnacki, and J.R. Wolpaw. Brain-computer interface (bci) operation: Optimizing information transfer rates. *Biological Psychology*, 63(3):237–251, 2003.