**Title**
Incorporating flash adjacency into the classifier for a language model-based P300 Speller



**Author**
Ma, Tianze

**Publication Date**
2023

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

# Incorporating flash adjacency into the classifier for a language model-based P300 Speller

A thesis submitted in partial satisfaction

of the requirements for the degree Master of Science

in Electrical and Computer Engineering

by

Tianze Ma

2023

ABSTRACT OF THE THESIS

Incorporating flash adjacency into the classifier

for a language model-based P300 Speller

by

Tianze Ma

Master of Science in Electrical and Computer Engineering

University of California, Los Angeles, 2023

Professor Jonathan Chau-Yan Kao, Chair

The P300 speller is a common brain–computer interface (BCI) application designed to allow patients with neuromuscular disorders such as amyotrophic lateral sclerosis (ALS) produce text output through the detection of P300 signals in their electroencephalogram (EEG) signals. The standard P300 speller relies on the detection of signals evoked by visual stimuli, usually consisting of rows and columns highlighted in a grid of characters. Since the visual field is substantially larger than these stimuli, adjacent flashes to the attend characters may cause false positive signals and lead to erroneous output. While previous work has tried to address this issue by limiting the number of adjacent stimuli, no attempts have been made to account for

these adjacency false positives in the classifier or utilize information from adjacent flashes to optimize the system. In this study we added a bias to the target character detection based on adjacent flashes and created a new probability model to improve the accuracy and speed of classification. We tested our adjacency classifier in both the standard P300 paradigm (we call it SWLDA method in the following content) and in conjunction with natural language processing. The new algorithm was evaluated offline on a dataset of 69 healthy subjects, which showed increases in speed and accuracy when compared to standard classification methods. On population level, LDA classifier shows a significant improvement, but the improvement for NLP is not significant. However, for some subjects in both algorithms the enhancement of information transfer rate is substantial. As for LDA classifier, 57.4% subjects' peak ITR increases of more than 5 bits per minute, 30.9% subjects' peak ITR increases more than 10 bits per minutes, and 8.8% subjects' peak ITR increases more than 15 bits per minutes. As for NLP, of the subjects, 21.7% had peak ITR increases of more than 5 bits per minute, 10.1% subjects' peak ITR increased more than 10 bits per minutes, and 5.8% subjects' peak ITR increased more than 15 bits per minutes. Therefore, incorporating adjacent fleshes can potentially provide a better communication system for some users.

The thesis of Tianze Ma is approved.

Corey Wells Arnold

William Hsu

William F Speier

Jonathan Chau-Yan Kao, Committee Chair

University of California, Los Angeles

2023

CONTENTS

List of Figures

# 1. INTRODUCTION

A person's ability to communicate can be impaired by the interruption of the signal transmission from the central nervous system to effector muscles. High brain stem injuries and motor neuron diseases such as amyotrophic lateral sclerosis (ALS) can cause such neural disorder. By detecting electrical signals from the brain and translating them into computer commands, Brain–computer interfaces (BCI) can restore some of this ability and provide a possible way for patients to communicate with the external world. With P300 speller, patients with motor neuron diseases are able to communicate with care workers and put forward their need in a timely manner, which can substantially enhance their life quality.

The P300 speller is an electroencephalogram (EEG)-based BCI device that uses electroencephalogram (EEG) signals to simulate keyboard input [1]. It can utilize P300 signal, which is a characteristic positive potential after a 300-millisecond delay from stimulus presentation[2]. When the experiment starts, a matrix composed of alphanumeric characters will be presented by the system. The user is asked to attend one character on the matrix and meanwhile the letters on the matrix flash randomly. According to the "oddball" paradigm, attending to the target character on the matrix will elicit the P300, given that the target character flashes relatively infrequently in the repeated stimuli. Then the traditional classification algorithm would take the brain response recorded by EEG as input and classify them into attended, and non-attended. Finally based on the classifier, the system will detect which character on the matrix was selected.

Traditionally, enhancing specific components of the P300 speller apparatus is the focus of the system optimization studies. For example, Allison et al. modified the matrix size, demonstrating that increasing the size of matrix improves the amplitude of the P300 signal [3]. Lu et al. evaluated the inter-stimulus-interval (ISI), suggesting a longer ISI translates to both a higher

online accuracy and higher characters per minute (CPM) rate [4]. Both Townsend et al. and Jin et al. developed novel flashing patterns, demonstrating significant improvements in bit rate and practical bit rate compared to the traditional row column paradigm (RCP) [5], [6]. In addition, Speier et al took advantage of exciting knowledge about the language domain to enhance the probability model and further enhance the decoding accuracy and speed [7].

While the P300 speller is designed to provide a means for communication, most attempts at system optimization have not considered the effect of adjacent flashes. Townsend's paper points out , "errors arise when flashes of non-target rows or columns that are adjacent to the target, attract the participant's attention, thereby producing P300 responses."[5] Participant's attention is attracted to adjacent flashes that appear in their peripheral vision. Given such phonemes, we predict that adjacent flashes will elicit brain signals which can be distinguished from P300 signal and non-P300 signal. However, existing analyses only classifies EEG responses into attended and non-attended, and therefore the false positive caused by flashes of adjacent rows and columns finally leads to the misdetection of the target letter.

To address the problem of false positive classification caused by adjacent flashes, we try to utilize the information from peripheral vision. We accomplished this task by creating a new class, adjacent, into the classification algorithm. Furthermore, we add a bias to the target character detection based on adjacent flashes and hypothesize that both system speed and accuracy can be improved.

## 2. Materials and Methods

### 2.1 Subjects and Data Collection

Data for offline analyses were obtained from 78 healthy volunteers with normal or corrected to normal vision between the ages of 20 and 50. G.usb amplifiers, active EEG electrodes, and electrode cap (Guger Technologies, Graz, Austria) were used to acquire data; the sampling rate was 256 Hz, the reference was the left ear; the ground was AFZ; and a bandpass filter of 0.1–60 Hz was used. A previously published configuration was used to set the electrode which consisted of 32 channels.[8] The system used a 36-character set, row and column flashes. The stimulus duration was 100 ms and the interstimulus interval was 25 ms. The stimulus onset asynchrony time was 125 ms.

All subjects were asked to spell twenty characters, “JUMP OVER “and “BROWN FOX ”. For each subject, EEG signals corresponding to the first 10 characters were the test set and the rest would be the training set.  For each character we recorded 120 stimulus, and our dynamic stopping algorithm would determine how many flashes we used to decode the character.

### 2.2 Incorporating Adjacency into P300 Classification

The standard P300 classification uses two classes: target and non-target, however it has been shown that stimuli adjacent to the target can also elicit event-related potentials (ERP) through peripheral vision. For example, if the target character is “P”, any flash of row or column containing “I”, ”J”, “K”, ”O”, “Q”, ”U”, “V”, ”W” can also elicit ERP (Figure 1). One approach to address this problem has been to try to minimize adjacent stimuli, but it is impossible to avoid completely. The reason is that it is  impossible to distinguish between characters without providing differential stimuli (i.e., flashing one and not the other), so adjacent characters must

be flashed in order to be possible outputs.

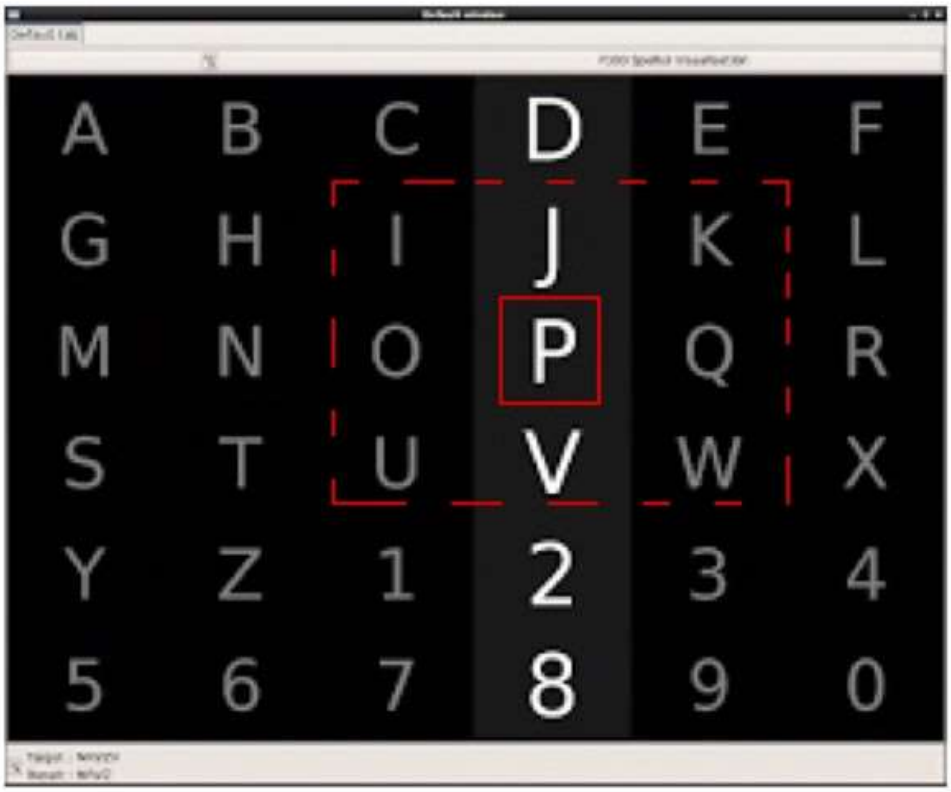


Figure 1： Example to illustrate the adjacent letters and adjacent stimulus.

Stimulus responses show a clear difference between target and non-target, which is the basis for p300 classification. Similarly, when looking at adjacent stimuli, they also demonstrate a different distribution from either of the two traditional classes. Figure 2 shows the histogram of the three classes when we feed stimulus responses into linear discriminant analysis. We want to utilize this fact in our classification.

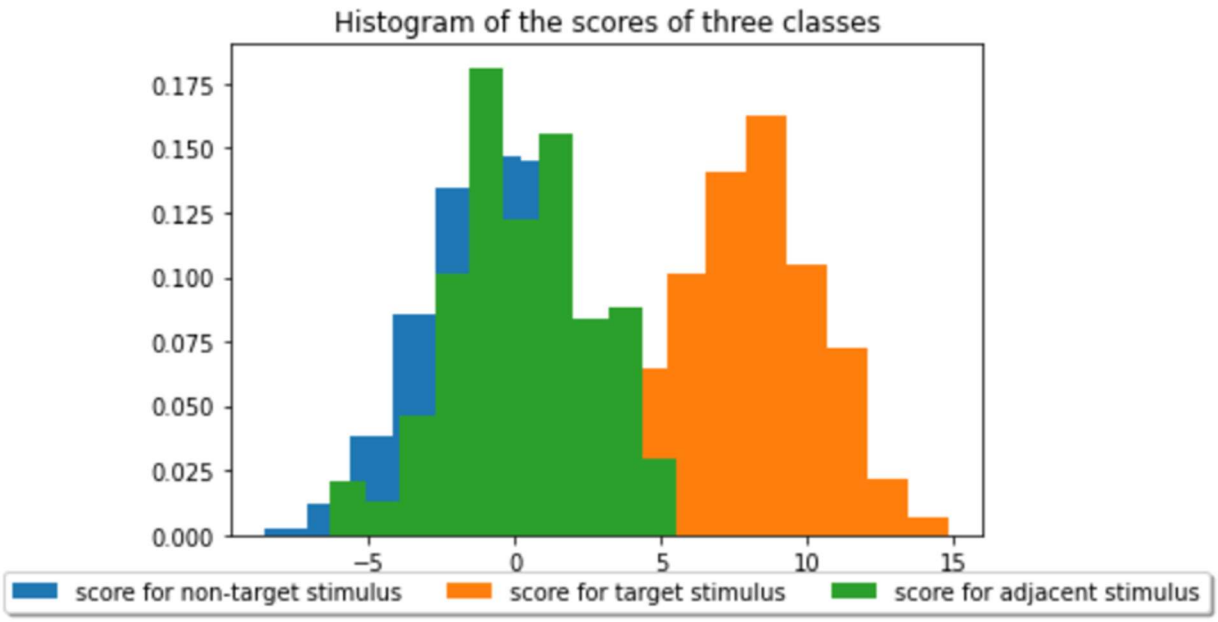


Figure 2： The histogram of the three classes when we feed stimulus responses into linear discriminant analysis.

We create a 3-class classification task to represent this underlying representation and expect that doing so can mitigate the effect of false positives from adjacent flashes and lead to better classification performance.

## 2.3 Dynamic Method with SWLDA Classifier

Stepwise Linear Discriminant Analysis (SWLDA) is a classification algorithm that selects a set of signal features to include in a discriminant function [9]. The signals in the training set are then assigned labels based on stimulus type. There are three classes: those corresponding to flashes containing the attended character, those without the attended character and not adjacent to the attended character, and those adjacent to the attended character. The traditional method does not distinguish between the second and the third class, and only classifies the

signal into two classes. The new method, however, has three classes. We feed training data with its corresponding labels into SWLDA, and after fitting with the training data set, SWLDA will generate a weight vector w. The score for the ith flash for character t in the sequence in the training data set is computed as the dot product of the feature weight vector.

$$Y_{ti}^{training} = w.Z_{ti}^{training}$$

After computing the scores for each flash in the training set, we make an assumption of the scores' distribution. In our study, we tested on 3-d Gaussian model, 2-d Gaussian model and 1-d Gaussian model. Among these assumptions, 1-d gaussian distribution performed the best.

The scores for ith flash for character t in the sequence in the test set, $Y_{ti}^{testing}$, was then computed as the dot product of the feature weight vector, w, with the features from that trial's signal, zit.

$$Y_{ti}^{tes} \quad = w.Z_{ti}^{testing}$$

For one dimensional gaussian distribution model, the probability density function, $f(x_t)$, for the likelihood probability can then be computed,

$$f\left(Y_{ti}^{testing}|x_t\right) = \begin{cases} \frac{1}{\sqrt{2\pi\sigma_0^2}} e^{\frac{1}{2\sigma_0^2}(Y_{ti}^{testing}-\mu_0)^2}, if\ x_t \in A_{ti} \\ \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{\frac{1}{2\sigma_1^2}(Y_{ti}^{testing}-\mu_1)^2}, if\ x_t \in Adj_{ti} \\ \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{\frac{1}{2\sigma_1^2}(Y_{ti}^{testing}-\mu_1)^2}, if\ x_t \notin A_{ti}\ and\ x_t \notin Adj_{ti} \end{cases} \quad (1)$$

, where $A_{ti}$ is set of characters illuminated for the ith flash for character t in the sequence

$Adj_{ti}$ is set of characters adjacent to the illuminated characters for the ith flash for character t in the sequence

For the traditional method, $\mu_0$, $\sigma_0$, $\mu_1$and $\sigma_1$, n are the means and standard deviations of the distributions for the attended and unattended flashes, respectively.

For the new method, $\mu_0$, $\sigma_0$, $\mu_1$, $\sigma_1$, $\mu_2$, and $\sigma_2$, n are the means and standard deviations of the distributions for the attended, adjacent and unattended flashes, respectively.

As for multivariate gaussian model:

$$f\left(Y_{ti}^{testing}\middle|x_t\right) = \begin{cases} (2\pi)^{-\frac{k}{2}}\det(\Sigma_0)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(x-\mu_0)^T{\Sigma_0}^{-1}(x-\mu_0)\right), if\ x_t \in A_{ti} \\ (2\pi)^{-\frac{k}{2}}\det(\Sigma_1)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(x-\mu_1)^T{\Sigma_1}^{-1}(x-\mu_1)\right), if\ x_t \in Adj_{ti} \\ (2\pi)^{-\frac{k}{2}}\det(\Sigma_2)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(x-\mu_2)^T{\Sigma_2}^{-1}(x-\mu_2)\right), if\ x_t \notin A_{ti}\ and\ x_t \notin Adj_{ti} \end{cases} \quad (2)$$

For the traditional method, $\mu_0$, $\Sigma_0$, $\mu_1$and $\Sigma_1$ are the means and covariance matrices of the distributions for the attended and unattended flashes, respectively. K is the dimensional of the gaussian model.

For the new method, $\mu_0$, $\Sigma_0$, $\mu_1$, $\Sigma_1$, $\mu_2$, and $\Sigma_2$, are the means and covariance matrices of the distributions for the attended, adjacent and unattended flashes, respectively. K is the dimensional of the gaussian model.

Although it has been shown that consecutive flashes are correlated [10], we assume that each flash's score was drawn independently from one of these distributions. Then we created two vectors, row probability and column probability, to record the accumulated probability from

flashes. The row probability and column probability are both initialize as [1,1,1,1,1,1], and are updated according to the following algorithm for each flash:

1.The probability of containing the attended character will be multiplied into the corresponding row or column.

2.For the new method, the probability of adjacent to the attended character will be multiplied into adjacent row or column.  For the traditional method, The probability of those without the attended character and not adjacent to the attended character will be multiplied into adjacent row or column.

3. The probability of those without the attended character and not adjacent to the attended character will be multiplied into the rest rows or columns.

4. Normalize the row probability and column probability.

Given an example for the traditional algorithm, if the classification probability of a row 2 flash is attended: 80%, non-attend:20%. Then the row probability would be updated as [20%,80%,20%,20%,20%,20%]. After normalization, the probability array would be updated as [11.1%,44.4%,11.1%,11.1%,11.1%].

Given an example for the new algorithm, if the classification probability of a row 2 flash is attended: 80%, non-attend:15% and adjacent:5%. Then the row probability would be updated as [5%,80%,5%,15%,15%,15%]. After normalization, the probability array would be updated as [3.70%,59.26%,3.70%,1.11%,1.11%].

The probability of each character is computed as the product of corresponding normalized row and normalized column probability. A threshold probability, PThresh, is then set to determine when a decision should be made. The program will keep take new flashes to update probability

matrix until either max(P (xt|yt, xt−1,...,x0) )> Pthresh or the number of sets of flashes reaches 120(number of total flashes for one character). The number of flashes used and the decoding accuracies achieved are recorded for values of Pthresh from 0 to 1.

## 2.4 NLP Method

We incorporate language models into the P300 speller using particle filtering (PF). The idea is from Speier et. [7] The particle filter classifies P300 signals using a Bayesian process model. From a probabilistic graph, the prior probabilities are determined, and progression through the model is estimated using Sequential importance resampling (SIR).

### 2.4.1 Language Corpus

Language models are built based on text representing the target output domain. Generally, large corpora of documents are used to avoid specificity. In this study, we used the Brown English corpus.[11] The Brown corpus consists of documents that contain a total of 1,014,312 words and it is a sufficient representation of the language for our purposes. However, it is not necessary and any other corpus that large enough can be used to build the language model as well. To simplify the corpus, we preprocess it by removing punctuation and stemming before we build the language model. In addition, in the language model, case is not distinguished.

### 2.4.2 Probabilistic Graph Language Model

A probability graph was created to model the English language by creating states for every substring that starts a word. For example, the word “for” would result in three states: ‘f,’ ‘fo,’ and ‘for’. The root node, $x_0$, is the start state, which corresponds to a blank string. Each state links to

all states that appends one more character than itself. For example, “f” links to “fa”, “fe”, “fo” etc. For each word, we use an “_” to denote the word ends

States which represent completed words contain links back to $x_0$, and a new word will begin afterwards. The state ‘for,’ for instance, links to the terminal state ‘for_’ which includes the valid English word ‘for’ and a space indicating the intent to start a new word. The state ‘for’ also contains links to nonterminal states such as ‘fort’ because it begins other English words in addition to being a complete word.

The transition probabilities are determined by the relative frequencies of words starting with the states’ substrings in our language corpus. [11]

$$P\ (x_{0:t}|x_{0:t-1})=\frac{c(x_1,x_2\ldots,x_{t-1},x_t)}{c(x_1,x_2\ldots,x_{t-1})}, \quad (3)$$

where c ($x_1$, $x_2$) denotes the number of occurrences of a word that starts with the string ‘$x_1x_2$’ in the corpus. p($x_{0:t}|x_{0:t-1}$)is the fraction of words, $x_{0:t}$, in the copus that begin with character ‘$x_{0:t-1}$’. Given an example in the above content, P ("$for$"|"$fo$") equals to the number of occurrences of a word that starts with the string ‘$for$’ in the corpus divided by the number of occurrences of a word that starts with the string ‘$fo$’ in the corpus.

Similarly, the probability that a word ends and the model transitions back to the root, $x_0$, is the ratio of the number of occurrences of complete words consisting of a string to the total number of occurrences of words beginning with that string

$$P\ (x_{0:t}|x_{0:t-1})=\frac{c(x_1,x_2\ldots,x_{t-1},\text{‘\_’})}{c(x_1,x_2\ldots,x_{t-1})}, \quad (4)$$

Where c ($x_1$, $x_2$, ‘_’) denotes is the number of occurrences of the complete word ‘$x_1x_2$’ in the corpus. In the example above, c (‘f’, $'o'$, $'r'$, ‘_’) denotes the number of occurrences of the complete word ‘for’ in the corpus.

2.4.3 Particle Filter

The Particle filter method estimates the probability distribution over possible output strings by sampling a batch of possible realizations of the model called particles. Each of these particles moves through the language model graph independently, based on the probabilities of transition. Realizations with low probability are removed and replaced by more likely realizations when resampling the particles based on weights derived from the analysis of EEG signals. The probability distribution of the possible output strings is estimated by computing a distribution of the particles after they have moved through the model graph. [7]

The algorithm is initialized by instantiating a set of J distinct particles and our study set J equals to 1,000 (Figure 3).

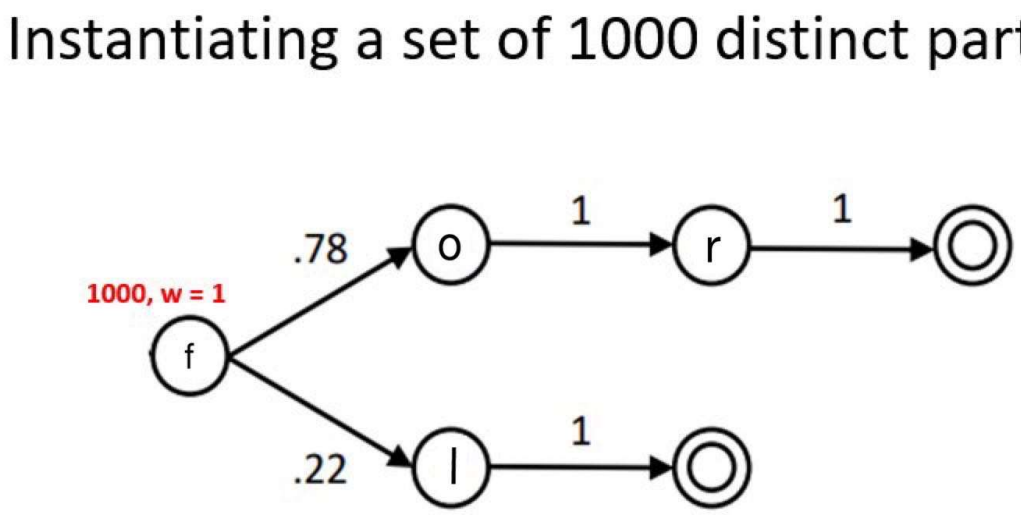


Figure 3： Demonstration of the step of initialization of a set of particles in particle filtering.

Each particle j has three attributes: history, node, and weight. History, $x_{0:t}^{(j)}$, is the string consisting of the particle's state history. Node, $n_j$, is the particle's current state. Weight, $w_j$ is the particle's probability weight. When the system begins, a set of particles is generated and each is associated with the root node, $x_0$, with an empty history and a weight equal to 1. At the start of a new character t, all particles would be projected to one of all the linked states based on the distribution defined by the language model's transition probabilities from the particle's history $x_{0:t-1}^{(j)}$.

$$x_j^{0:t} \sim \text{p}\,(x_{0:t}|x_{0:t-1}^{(j)}), \tag{5}$$

where p $(x_{0:t}|x_{0:t-1}^{(j)})$ is provided by the language model (equation (4)).

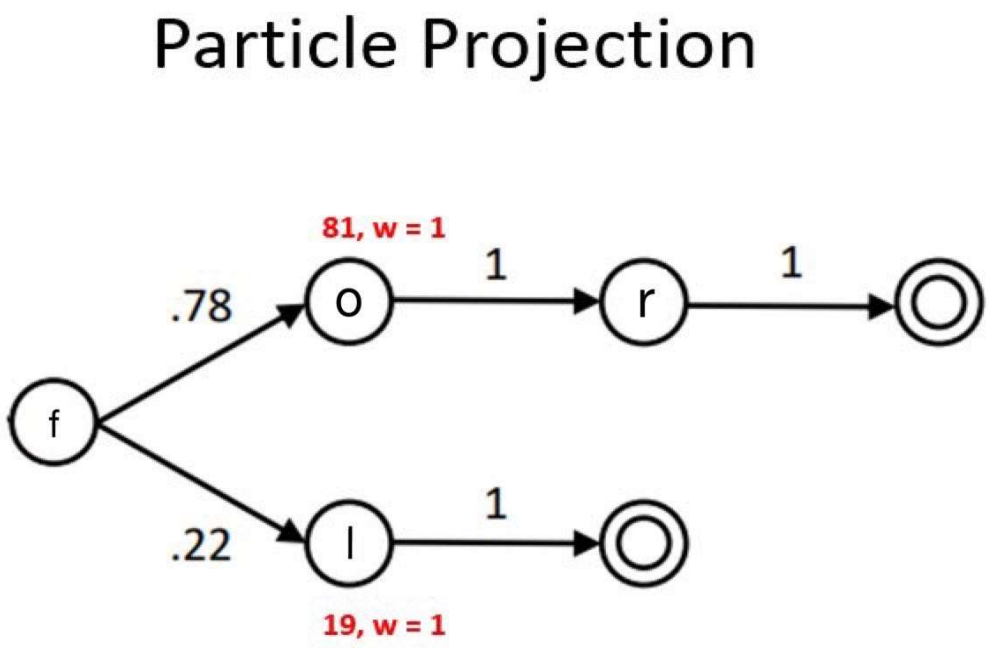


Figure 4： Demonstration of the step of particle projection in particle filtering.

After the projection (Figure 4), the state that the particle is projected to would be append to the attribute history, $x_j^{0:t}$ , and the node would be changed to $x_t^{(j)}$. After each stimulus response, the probability weight is computed for each of the particles.

$$w_j \propto p\left(x_t^{(j)}\right) \propto \Pi_k f(x_t)$$, k is iterated through all flashes.

where $f(x_t)$ is computed as in equation (1) or (2) based on the distribution model. The weights are then normalized, and the probability of the current character is found by summing the weights of all particles that end in that character. For each time the weight is updated, the summation of weight of all the particles in the same state would be computed. (Figure 5)

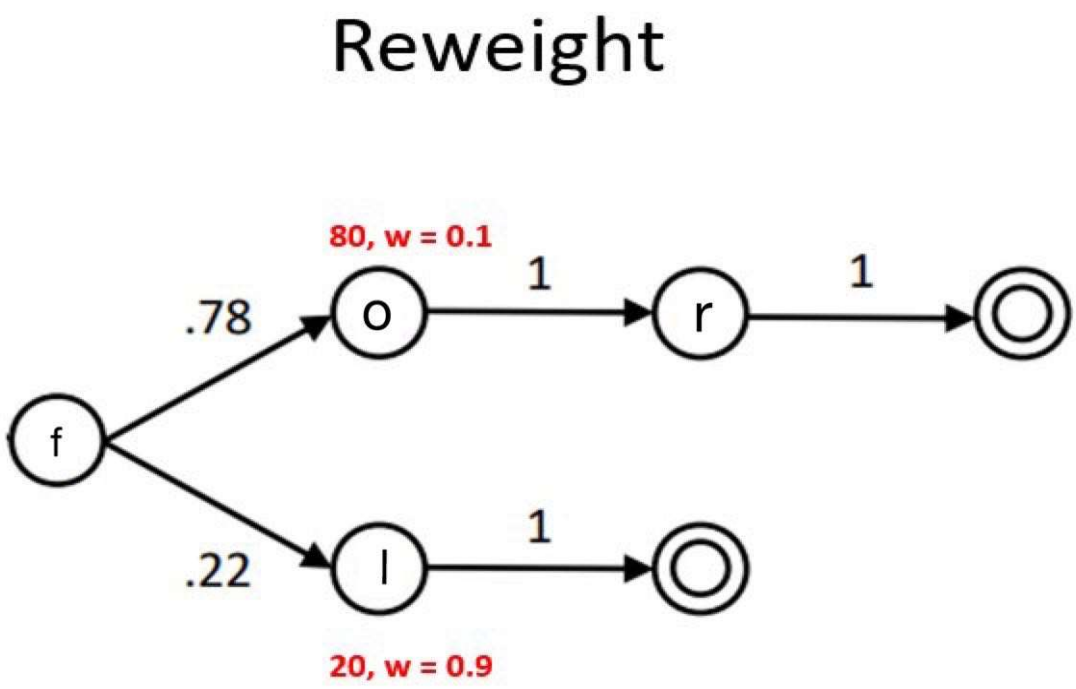


Figure 5: Demonstration of the step of reweight in particle filtering.

If the maximum summation in one state is above a threshold, the particle with the maximum weight is selected and its history is used as the output text. Once the threshold is achieved, all particles would be resampled. A multinomial distribution is created over the set of particles where each particle's probability is proportional to its weight. A new set of J particles is then sampled from this distribution with replacement. This set of particles is then used in the next time step. (Figure 6)

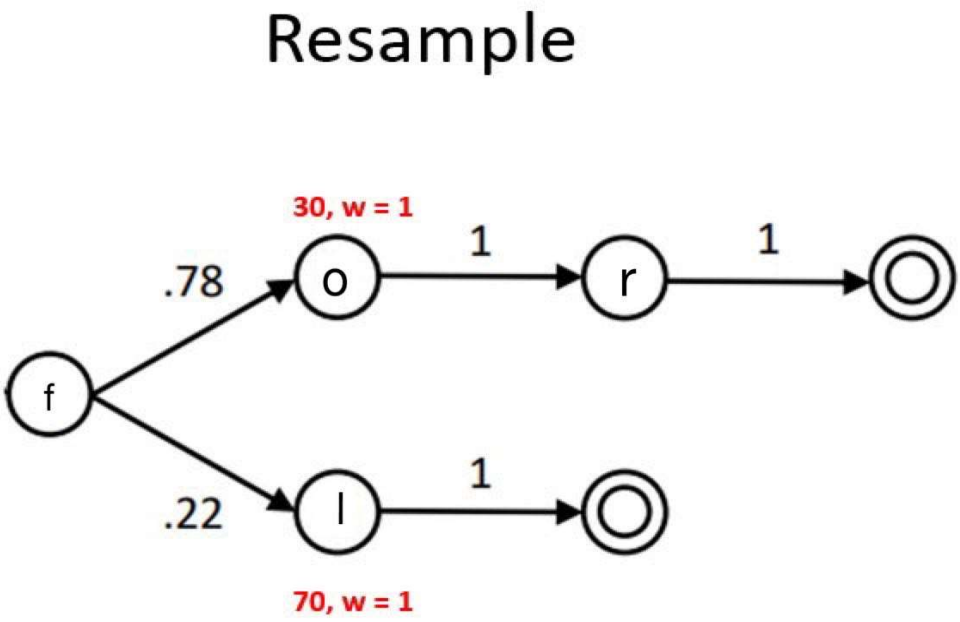


Figure 6: Demonstration of the step of resample in particle filtering.

After the resample is finished, we will start to repeat the decoding process of the next character by projecting all particles again.

## 3. Evaluation

We take two factors into account when evaluating the BCI system: the capability of the system to achieve the desired result and the amount of time required to achieve that result. The efficacy

of the system can be measured as the accuracy of selection, which we defined as the percentage of decoded characters that matched the target string. The speed of the system was measured by how many flashes on average were presented by the system before it returns the output characters. Under the premise that other conditions remain unchanged, the used flashes are less, the speed of the system is higher. In order to consider the factors of speed and accuracy, we compute information transfer rate (ITR) for each method.

$$ITR = \frac{B}{T}, B = \log N + P\ logP + (1-P) log \frac{1-P}{N-1} \quad (6)$$

, where B is the information transferred in bites per trial, N is the number of targets, P is the classification accuracy. In our study, N = 36

To evaluate if the enhancement between traditional method and our method is statistically significant, we run the Wilcoxon signed-rank test across the whole subject set and the alpha value of 0.05 is used.

## 4. RESULTS

### 4.1 Traditional SWLDA Classifier vs. SWLDA classifier with adjacent flash results for accuracy vs. time

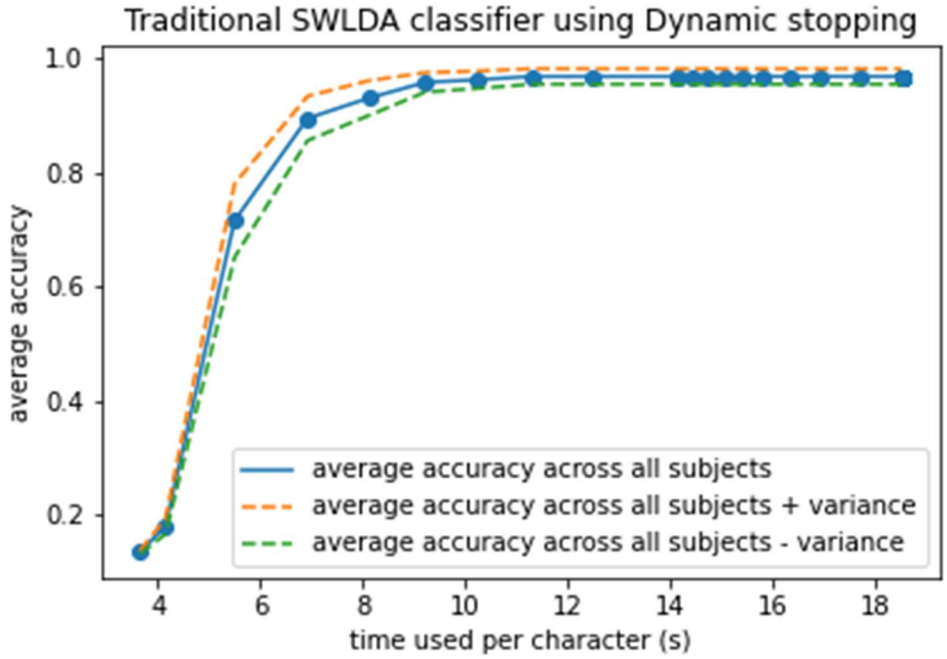


Figure 7 Average accuracy across all subjects for the classifier using Dynamic stopping without adjacent flashes. Dotted lines correspond to the average +/- variance.

From

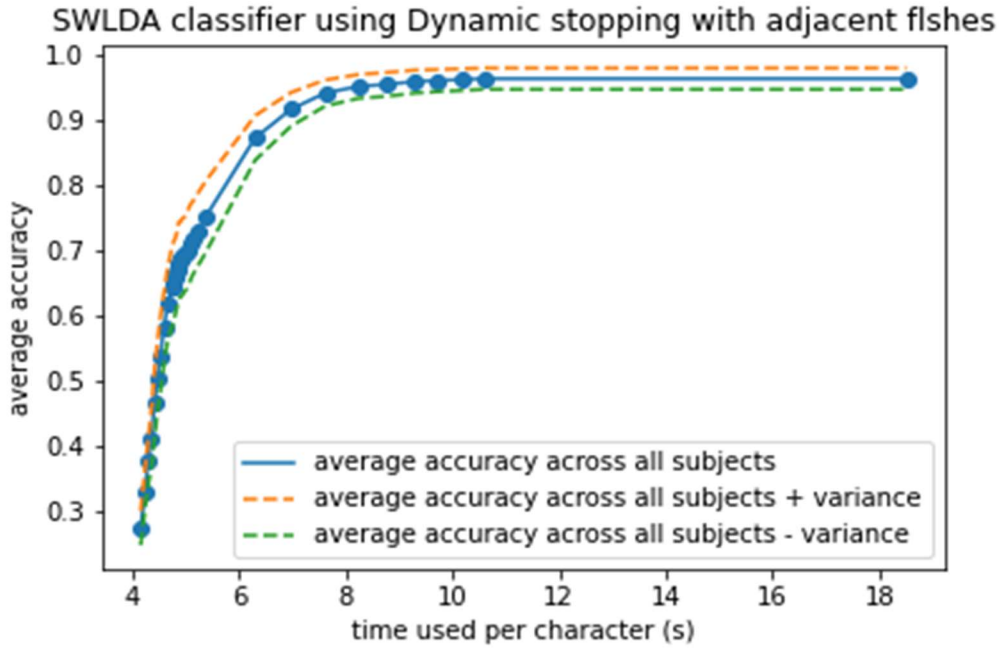

Figure 8: Average accuracy across all subjects for the classifier using Dynamic stopping using adjacent flashes. Dotted lines correspond to the average +/- variance.

The curves in Figure 7 and Figure 8 are in the same shape. At the beginning, the accuracy increases as the time used per character increases. It achieves an asymptotic maximum value at some point. As for the traditional SWLDA method, the asymptotic maximum value is achieved when time equals to 9 and as for the SWLDA method, the asymptotic maximum value is achieved when time equals to 10. In addition, the asymptotic maximum values for the two method are almost the same (96.9%).

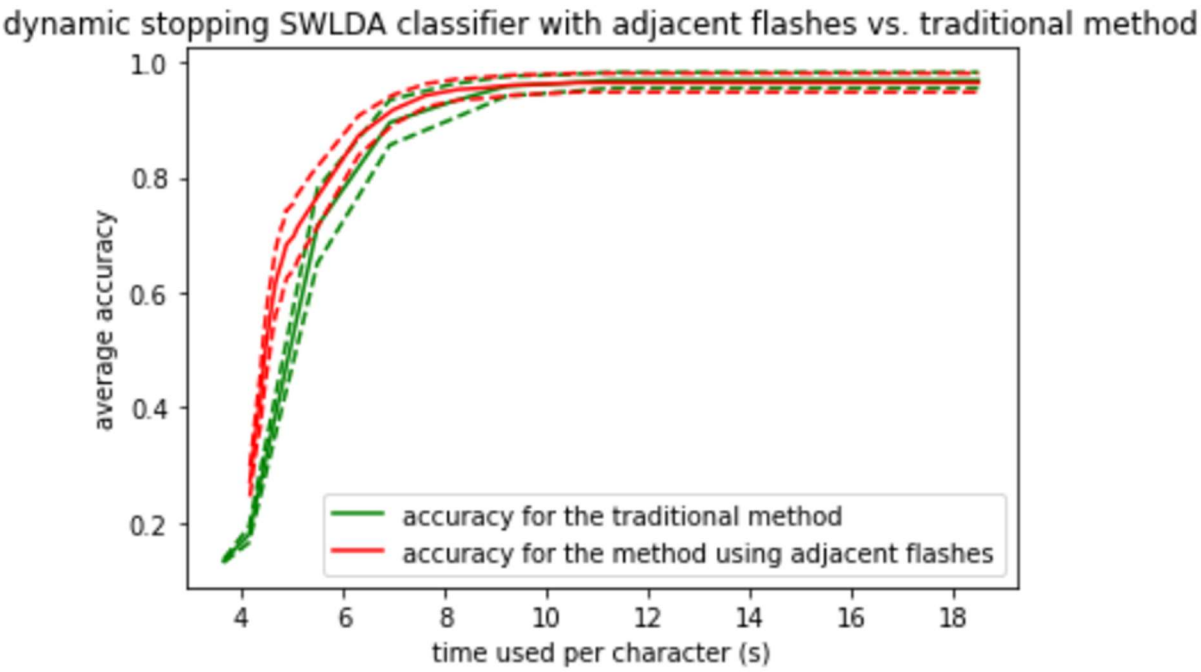


Figure 9: Average accuracy across all subjects for the classifier using Dynamic stopping using vs. not using adjacent flashes in the same figure.

From Figure 9, across all the time given, the accuracy of LDA using adjacent flashes is always higher than the accuracy of traditional methods.

ITR over different thresholds

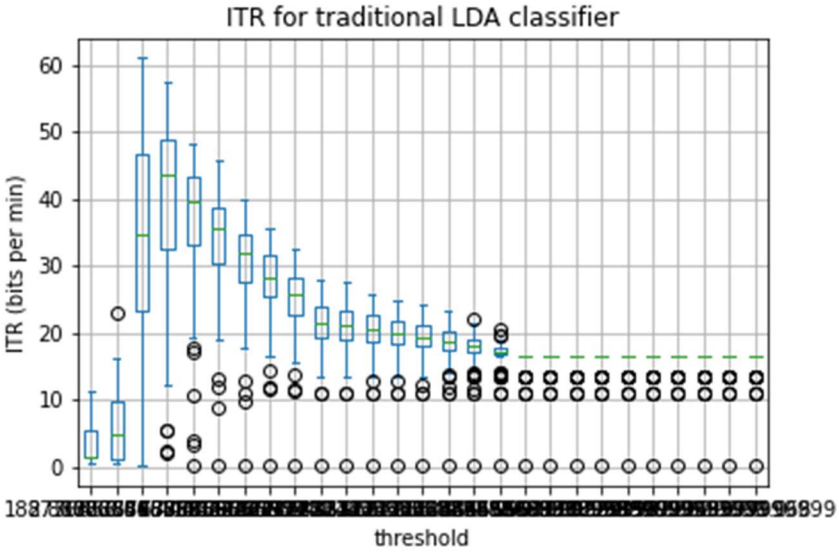


Figure 10: Boxplot of ITR across all subject set using traditional LDA classifier.

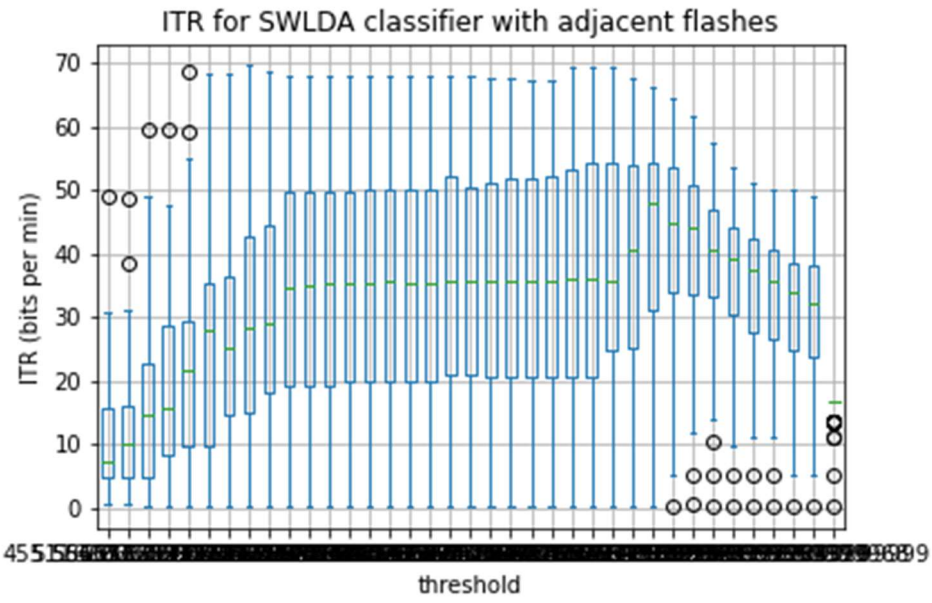


Figure 11: Boxplot of ITR across all subject set using LDA classifier using adjacent flashes.

From the box plot of ITR (Figure 10 and Figure 11), LDA using adjacent flashes has higher Peak ($Peak_{traditional} = 43.74, Peak_{adjacent} = 47.76$). In addition, the difference of maximum ITR across all the subject set is statistically significant. (P value for the t test: $P_{global} = 2.94e-4, P_{individual} = 3.66e-13$).

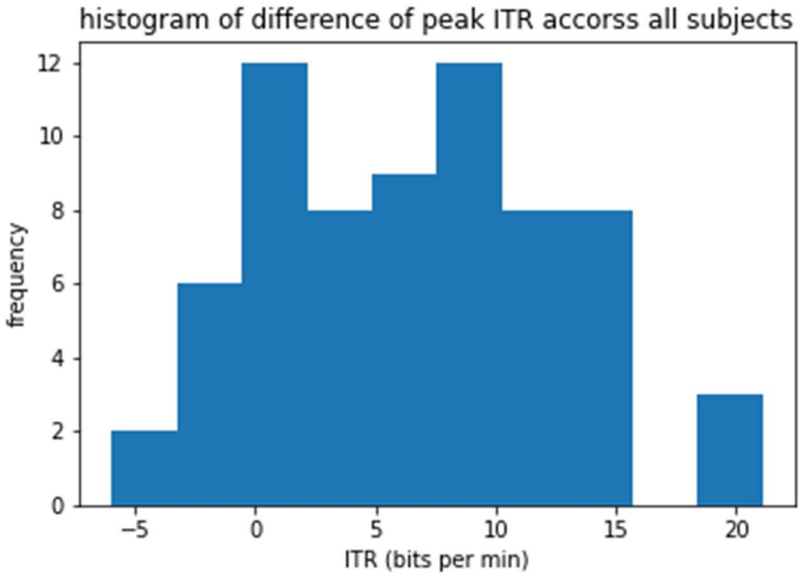


Figure 12: Histogram of difference of peak ITR across all subjects between the traditional method and the new method.

Figure 12 gives a histogram which shows a larger population who has positive difference of ITR than the negative. There are some subjects whose peak ITR increase and almost same amount of subjects whose ITR decrease. Of the subjects, 57.4% had peak ITR increases of more than 5 bits per minute, 30.9% subjects' peak ITR increases more than 10 bits per minutes, and 8.8% subjects' peak ITR increases more than 15 bits per minutes.

4.2 Traditional NLP Model vs. NLP Model Using Adjacent Flashes.

results for accuracy vs. time

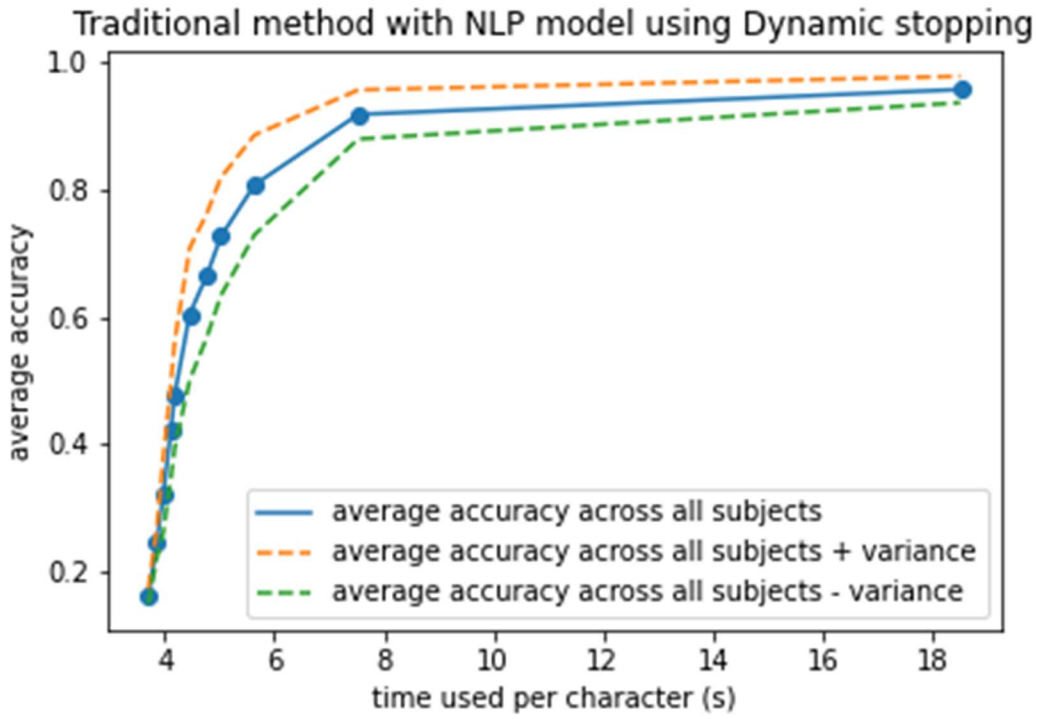


Figure 13: Average accuracy across all subjects for the NLP method using Dynamic stopping without adjacent flashes. Dotted lines correspond to the average +/- variance.

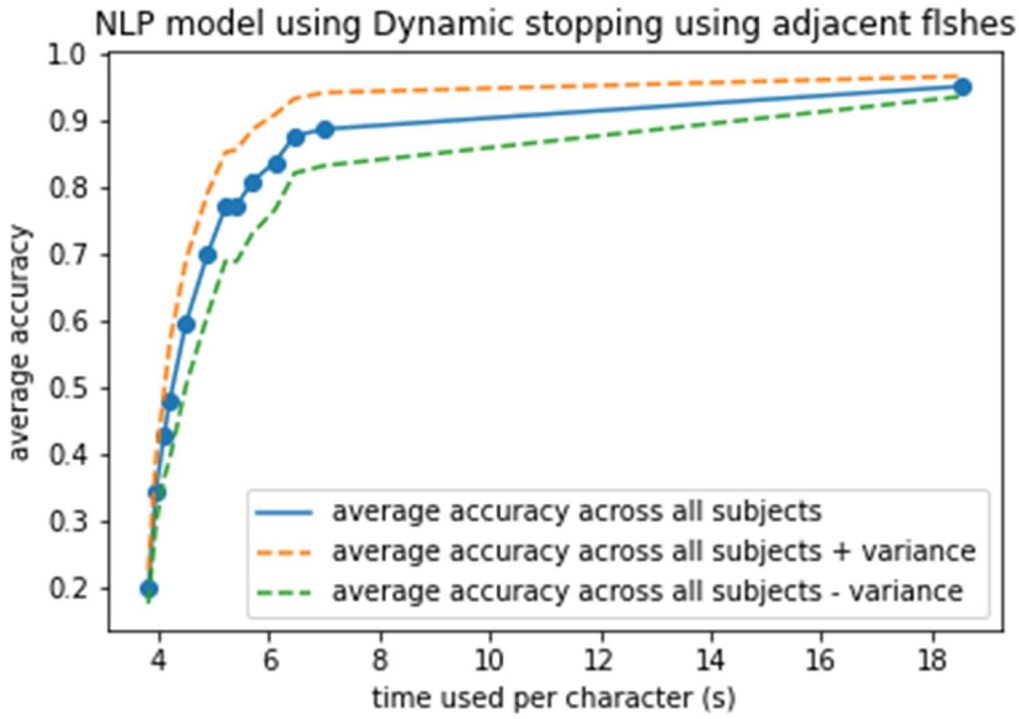


Figure 14: Average accuracy across all subjects for the NLP method using Dynamic stopping using adjacent flashes. Dotted lines correspond to the average +/- variance.

The curves in Figure 13, Figure 14 are in the same shape. At the beginning, the accuracy increases as the time used per character increases. It achieves an asymptotic maximum value at some point. The asymptotic maximum value is achieved at almost the time for the both method (9.5). In addition, the asymptotic maximum values for the two methods are also almost the same (94.9%).

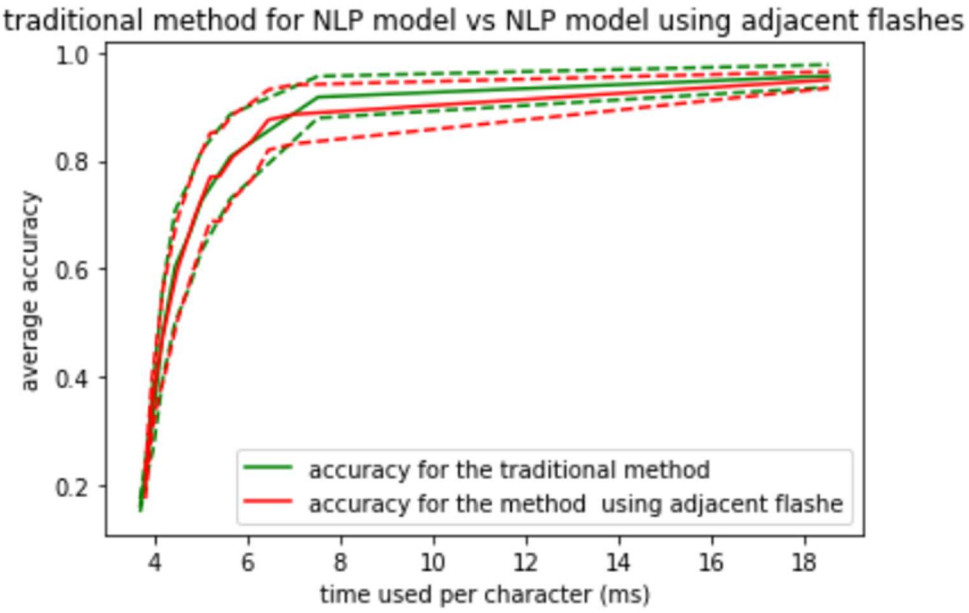


Figure 15: Average accuracy across all subjects for the classifier using Dynamic stopping using vs. not using adjacent flashes in the same figure.

From Figure 15, Across all the time given, the accuracy of LDA using adjacent flashes is not significantly higher than the accuracy of traditional method and the two curves with their +/- variance is almost overlapped.

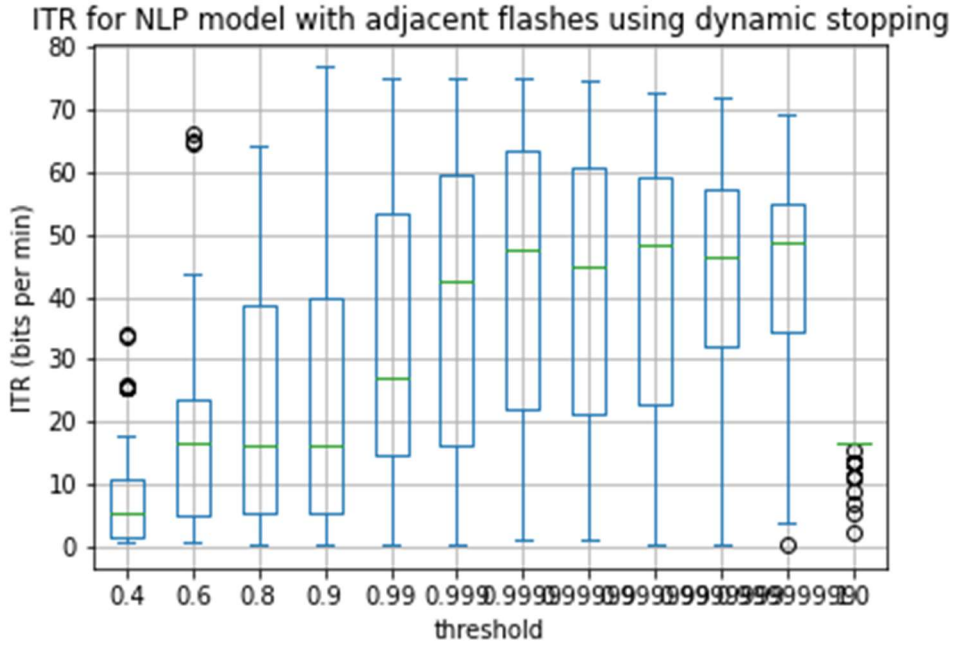


Figure 16: Boxplot of ITR across all subject set using traditional NLP method.

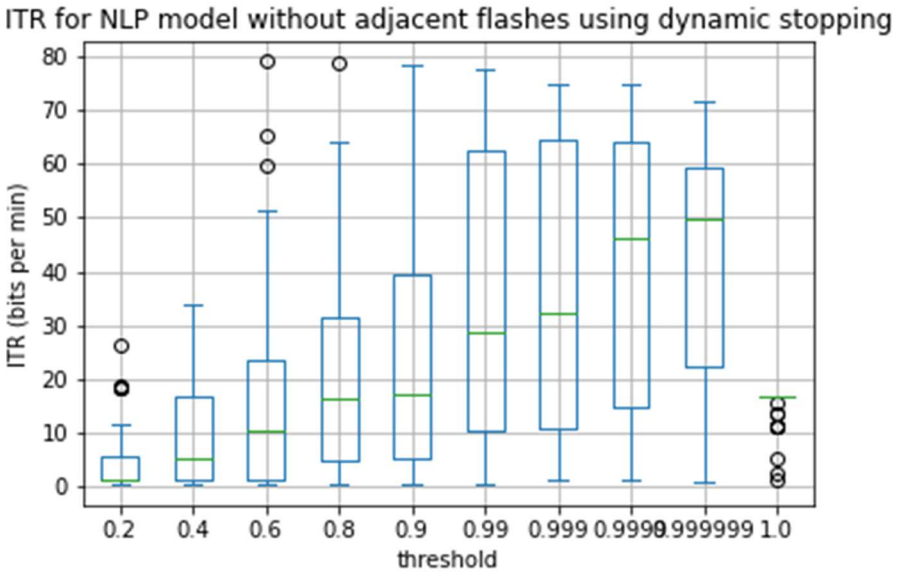


Figure 17: Boxplot of ITR across all subject set using NLP method with adjacent flashes.

From the box plot of ITR (Figure 16 and Figure 17), traditional LDA has higher Peak ITR for NLP model($Peak_{traditional} = 49.67, Peak_{adjacent} = 48.79$). In addition, the difference of maximum ITR across all the subject set is not statistically significant. (P value for the t test: $P_{global} = 0.876, P_{individual} = 0.0822$).

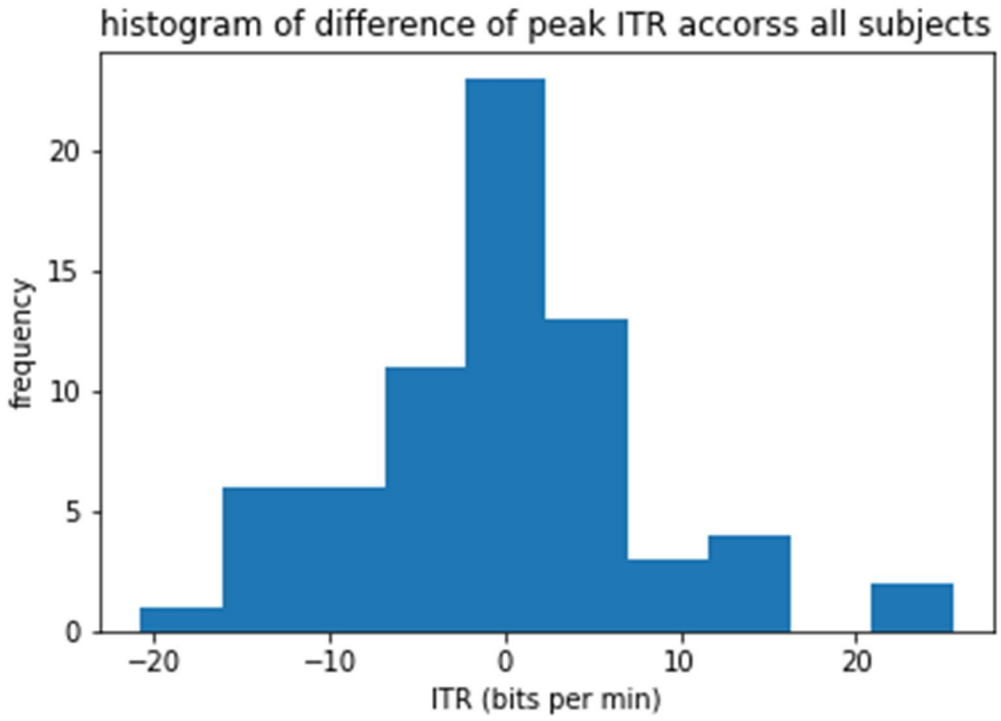


Figure 18: Histogram of difference of peak ITR across all subjects between the traditional method and the new method using NLP method.

Figure 18 gives a roughly symmetrical histogram. There are some subjects whose peak ITR increase and almost same amount of subjects whose ITR decrease. Of the subjects, 21.7% had peak ITR increases of more than 5 bits per minute, 10.1% subjects' peak ITR increased more than 10 bits per minutes, and 5.8% subjects' peak ITR increased more than 15 bits per minutes.

## 5. DISCUSSION

Our results show that adjacent flashes can improve the performance of P300 speller from the perspective of the whole data set. The median ITR is enhanced from 43.74 bit per minute to 47.76 bit per minute. Of the subjects, 57.4% had peak ITR increases of more than 5 bits per minute, 30.9% subjects' peak ITR increases more than 10 bits per minutes, and 8.8% subjects' peak ITR increases more than 15 bits per minutes. The results of NLP model are different from the results of SWLDA. We do not see a significant enhancement after incorporating adjacent flashes. In addition, the peak median ITR for the two algorithms are also very close to each other ($Peak_{traditional} = 49.67, Peak_{adjacent} = 48.79$). Of the subjects, 21.7% had peak ITR increases of more than 5 bits per minute, 10.1% subjects' peak ITR increased more than 10 bits per minutes, and 5.8% subjects' peak ITR increased more than 15 bits per minutes.

Adjacent flashes incorporate additional information from stimulus in peripheral vision and enable the speller to make faster decisions with higher accuracy. However, the enhancement for different subject varies. One potential reason is how easily the user is disturbed by the adjacent flashes. If the user is hardly disturbed by the adjacent flashes, the feature of neural response to adjacent flashes is close to non-attended flashes. Because of that, misclassification between adjacent and non-attend flashes happens. Though the algorithm to compute probability for character we designed in this study will not be harmed by this kind of misclassification, distinguishing adjacent flashes will not provide significant enhancements to the accuracy of the system in this case. However, as for NLP model method, the reason why we do not see much improvement of the new methods might because the Prior probability given by the NLP model already bias the probability distribution of all letters enough, and the difference made by adjacent flashes is very limited.

While the results of our study may not be practically significant for all the subjects, it shows that 8.8% of the subjects' peak ITR increases more than 15 bits per minutes for LDA dynamic method

and 5.8% of the subject's' peak ITR increases more than 15 bits per minutes for method with NLP model. For these subjects, the enhancement is significant in practice. Therefore, we think it is worthwhile to determine what kind of subject can be significantly benefited. Our lab is working with ALS patients who cannot move their eyes easily and therefore it is hard for them to concentrate on the letter on the screen. We hypothesize that the system using adjacent flashes will benefit ALS patient's more than healthy subjects, and the enhancement can be practically significant.

There is another problem, we find in PF algorithm, words with a small probability are likely to be mis-decoded. For example, "fox" is likely to be decoded as "for". In addition, since the PF algorithm is not able to make all corrections automatically one mistake would cause sequential errors. For instance, after the "x" in "foxes" is decoded as "r", the whole word "foxes" would likely be decoded as "force". Another solution is to build a Customized language model for each individual subject. We can record a subject's daily language and build a language model which has probability bias based on it. It can enhance the accuracy of the common word the subject uses, and therefore enhance the accuracy of the system overall.

Our research is all based on retrospective experiments so we cannot let the users offer their feedback. One future direction is to transform the decoding algorithm to online setting and let the user give feedback on time to correct the mis-decoded characters when using NLP model. It would slow the process down but can mitigate sequential errors.

In conclusion, incorporating adjacent flashes can enhance the performance of P300 speller for some subjects. The enhancement is more significant when not using NLP model. In addition, the enhancement for different subjects varies, and it is worth to determine which subject can be benefitted from this new method in the future research. For example, ALS patients are one

potential beneficiary group. Overall, P300 speller with adjacent flashes can provide better communication options for affected populations.

## 6. REFERENCES


1. Farwell LA, Donchin E. Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials. Electroencephalography and Clinical Neurophysiology. 1988;**70**(6):510–23.

2. Picton TW. The P300 wave of the human event-related potential. Journal of Clinical Neurophysiology. 1992;**9**(4):456–79.
3. Allison BZ, Pineda JA. Erps evoked by different matrix sizes: Implications for a brain computer interface (BCI) system. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2003;**11**(2):110–3.
4. Lu J, Speier W, Hu X, Pouratian N. The effects of stimulus timing features on p300 speller performance. Clinical Neurophysiology. 2013;**124**(2):306–14.
5. Townsend G, LaPallo BK, Boulay CB, Krusienski DJ, Frye GE, Hauser CK, et al. A novel p300-based brain–computer interface stimulus presentation paradigm: Moving beyond rows and columns. Clinical Neurophysiology. 2010;**121**(7):1109–20.
6. Jin J, Horki P, Brunner C, Wang X, Neuper C, Pfurtscheller G. A new p300 stimulus presentation pattern for EEG-based spelling systems. Biomedizinische Technik/Biomedical Engineering. 2010;**55**(4):203–10.
7. Speier, W., Arnold, C. W., Deshpande, A., Knall, J., & Pouratian, N. (2015). Incorporating advanced language models into the P300 speller using particle filtering. *Journal of Neural Engineering*, *12*(4), 046018. https://doi.org/10.1088/1741-2560/12/4/046018
8. Lu J, Speier W, Hu X, Pouratian N, “The effects of stimulus timing features on P300 speller performance”, Clinical Neurophysiology, Volume 124, Issue 2, 2013
9. Draper N and Smith H 1981 Applied Regression Analysis 2nd edn (New York: Wiley)

10. Citi L, Poli R and Cinel C 2010 Documenting, modeling and exploiting P300 amplitude changes due to variable target delays in Donchin's speller J. Neural Eng. 7 056006
11. Francis W and Kucera H 1979 Brown Corpus Manual (Providence, RI: Brown University)