Computers in Biology and Medicine 80 (2017) 97–106

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

MindEdit: A P300-based text editor for mobile devices

Amr S. Elsawy, Seif Eldawlatly*, Mohamed Taher, Gamal M. Aly

Computer and Systems Engineering Deptartment, Faculty of Engineering, Ain Shams University, 1 El-sarayat st, Abbassia, Cairo, Egypt

ARTICLE INFO

Keywords:
Brain-Computer Interface (BCI)
P300
Speller
Android
Emotiv

ABSTRACT

Practical application of Brain-Computer Interfaces (BCIs) requires that the whole BCI system be portable. The mobility of BCI systems involves two aspects: making the electroencephalography (EEG) recording devices portable, and developing software applications with low computational complexity to be able to run on low computational-power devices such as tablets and smartphones. This paper addresses the development of *MindEdit*; a P300-based text editor for Android-based devices. Given the limited resources of mobile devices and their limited computational power, a novel ensemble classifier is utilized that uses Principal Component Analysis (PCA) features to identify P300 evoked potentials from EEG recordings. PCA computations in the proposed method are channel-based as opposed to concatenating all channels as in traditional feature extraction methods; thus, this method has less computational complexity compared to traditional P300 detection methods. The performance of the method is demonstrated on data recorded from *MindEdit* on an Android tablet using the Emotiv wireless neuroheadset. Results demonstrate the capability of the introduced PCA ensemble classifier to classify P300 data with maximum average accuracy of 78.37 ± 16.09% for cross-validation data and 77.5 ± 19.69% for online test data using only 10 trials per symbol and a 33-character training dataset. Our analysis indicates that the introduced method outperforms traditional feature extraction methods. For a faster operation of *MindEdit*, a variable number of trials scheme is introduced that resulted in an online average accuracy of 64.17 ± 19.6% and a maximum bitrate of 6.25 bit/min. These results demonstrate the efficacy of using the developed BCI application with mobile devices.

1. Introduction

A brain-computer interface (BCI) is a system that provides a direct communication channel between the brain and the computer [1]. Such interface could help disabled subjects by providing them with means to compensate for their damaged peripheral nerves or muscles. BCIs acquire brain signals and decode them into commands to control an external device. One of the most important BCI applications is the P300 speller which allows the selection of characters on a virtual keyboard by analyzing recorded electroencephalography (EEG) activity [2]. The speller application is based on a brain signal pattern termed P300 potential which is characterized by a positive peak that appears about 300 ms following the presentation of a rare event to the user as shown in Fig. 1a [3]. The speller interface consists of a grid of characters as shown in Fig. 1b that are flashed randomly. The flashing represents the rare event that elicits the P300 potential when the flashed character matches the character the user is focused on. By synchronizing the flashing time with the brain signals, the character associated with the P300 potential can be identified.

One of the challenges that has been facing the practical use of BCI applications is the inconvenience associated with using traditional EEG recording systems manifested by the need for large amplifiers, using wired interfaces with the sensors and the need for applying conductive gel to enhance the quality of the recorded signals. However, recent advances in the development of EEG recording technology have resulted in commercially-available low-cost EEG recording headsets that can transmit the recorded signals wirelessly. Overcoming such hardware challenge has encouraged the development of practical BCI applications that can help people in daily life activities. Given the widespread use of mobile devices such as tablets and smartphones, developing BCI applications that can run on mobile devices could represent the future of BCI. Thanks to technology, the processing power of mobile devices is improving every day which could help running advanced signal processing algorithms. Examples of BCI applications that have been already developed to run on mobile devices include Neurophone which is a dialing application consisting of a 3×2 grid that enables users to call someone using the P300-paradigm. Such application can be considered the very first BCI mobile phone applica-

* Corresponding author.
E-mail addresses: aselsawy@eng.asu.edu.eg (A.S. Elsawy), seldawlatly@eng.asu.edu.eg (S. Eldawlatly), mohamed.taher@eng.asu.edu.eg (M. Taher), gamal.aly@eng.asu.edu.eg (G.M. Aly).

http://dx.doi.org/10.1016/j.compbiomed.2016.11.014
Received 6 August 2016; Received in revised form 24 November 2016; Accepted 26 November 2016

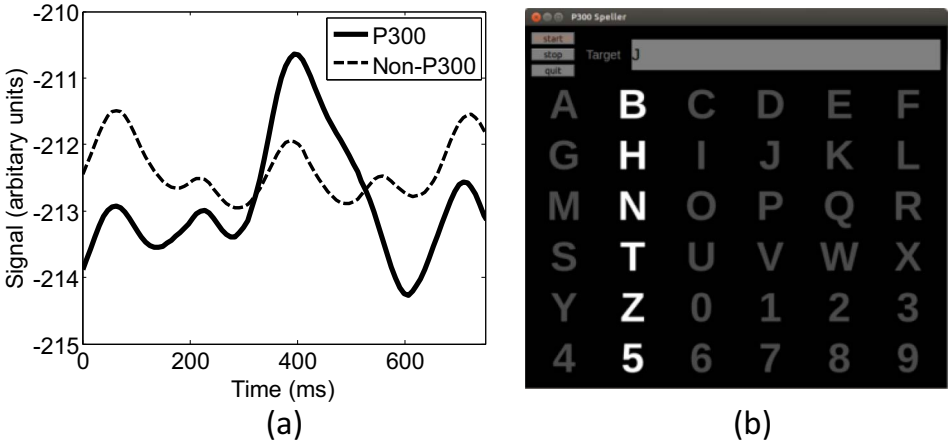

Fig. 1. (a) Mean responses of channel F3 for P300 and non-P300 potentials for one subject. (b) P300 speller interface showing a grid of characters in which rows and columns are intensified randomly while the subject is focusing on the character to spell.

tion [4]. However, Neurohpone is not a standalone application as it requires a personal computer to analyze the recorded EEG data and subsequently transmit the analyzed data to the mobile phone. Another example of BCI applications developed for mobile devices is the Smartphone brain scanner application which provides EEG imaging of brain activity [5]. Wang et al. proposed a dialing system that employs steady-state visual evoked potential (SSVEP) using online signal processing methods in both time- and frequency-domains [6]. Caruso et al. developed a P300-based prototype of an assistive system which allows the control of electrical devices at home using a tablet running Windows 7 [7]. Although these applications represent a good start, there are challenges that need to be addressed to use BCIs in real-life applications. For example, previous work did not address the computational complexity of the algorithms that run on mobile devices. In addition, to our knowledge, no study has addressed the design and implementation of complex mobile applications that involve large-grid P300 applications.

In this paper, we address the development of *MindEdit*; a P300-based text editor application for Android platforms based on the classical P300 speller paradigm. The general design of the system is shown in Fig. 2. We selected Android platform because it is an open-source mobile platform and it has most of the mobile devices market share which makes it the target for many developers to build their applications. To detect P300 patterns, we introduce a Principal Component Analysis (PCA) ensemble classifier algorithm. The proposed algorithm uses PCA to extract the most relevant features from the signals recorded on each channel. The extracted features of the i^{th} principal component of each channel are combined and used as an input to a classifier that attempts to discriminate between P300 and non-P300 potentials. Finally, the outputs of all trained classifiers are aggregated where the output of each classifier is weighted based on the importance of the corresponding principal component in representing the data. We demonstrate the efficacy of using such method on data recorded using the Emotiv wireless neuroheadset.

2. Methods

2.1. MindEdit Interface Implementation and Data Recording

We developed the interface of *MindEdit* using Qt C++ which is a cross-platform language that can run on Windows, Linux and Android platforms. For data recording, we utilized the decryption library of the smartphone brain scanner 2 open source project which is also developed using Qt C++ to acquire EEG data from the recording headset [8]. To synchronize the recorded data with the interface, we controlled the flashing instants of the interface by ensuring that the number of recorded samples represents the specified time. The interface of *MindEdit* text editor is shown in Fig. 3. In the *MindEdit* interface, we used symbols, in addition to characters, to represent actions that can be used to control the text editor. The actions of the developed text editor are summarized in Table 1. Text typed by the user using *MindEdit* interface can be manipulated using simple operations and can be saved to a file. In addition, the application also allows loading text from a file to the text editor screen. With exit and clear actions, we require a confirmation from the subject which is done by selecting ✓ after each detected action or otherwise the action is

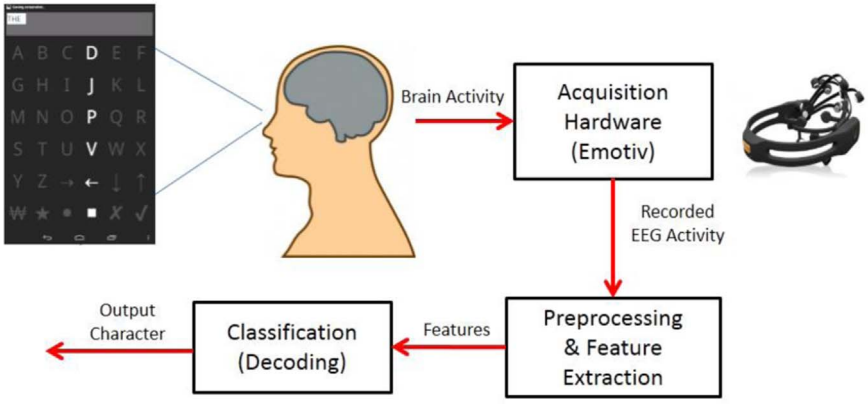

Fig. 2. General design of the *MindEdit* text editor. In this application, subjects are presented with a grid of symbols whose rows and columns are randomly intensified. EEG data is acquired using the Emotiv wireless neuroheadset. Recorded signals are pre-processed to filter noise and to extract relevant features. These features are then fed to a classifier whose function is to determine the flashed row/column and subsequently determine the selected symbol.

Fig. 3. *MindEdit* Android interface used to record our data. The interface consists of a 6×6 grid with characters and symbols that enable editing the typed text.

Table 1
MindEdit text editor actions.

Symbol	Action	Symbol	Action
→	Space	★	Clear all text
←	Backspace	■	Save text
↓	New line	•	Open saved text
↑	Delete one line	✗	Exit
₩	Delete word	✓	Confirm

ignored.

In our experiments, datasets from six healthy male subjects (named S1 through S6) were recorded using the commercially-available wireless Emotiv EPOC neuroheadset (Emotiv Systems Inc., San Francisco, USA). Subjects signed an informed consent approving the use of their data in this study. This headset has 14 electrodes at positions AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8 and AF4 based on the international 10–20 system. Recorded signals had a sampling rate of 128 Hz. Out of the six subjects, four subjects had previous experience with P300-based BCIs. In all experiments, a 6×6 grid was presented to the subject where each row and column was intensified for 100 ms. We used Inter-Stimulus Intervals (ISI) value of 300 ms. This interval was chosen based on our previous analysis on Linux [9]. A sequence of 12 intensifications (representing 6 rows and 6 columns of the grid) was repeated 10 times to constitute a single trial. Thus, each trial consisted of 120 intensifications. To train the proposed ensemble classifier, we recorded a training dataset for each subject consisting of 33 characters. The training dataset was recorded in one session with 60 s break after each 10 characters. For the online testing of *MindEdit*, we recorded an online testing dataset consisting of 20 characters. The testing dataset was recorded online in one session without breaks. Furthermore, we performed two online tests using variable trials to compare methods while in use. A time segment of 90 samples that corresponds to approximately 700 ms was then extracted for each epoch as determined by our previous analysis on Linux [9]. In all experiments, we used Nexus 7 Google Tablet with NVIDIA Quadcore CPU and Jellybean Android 4.1.2 operating system.

2.2. Preprocessing

To eliminate any across-channels noise, we first filter recorded signals using the common average reference spatial filter [10]. In this filter, the mean of signals recorded on all channels is subtracted from the signal recorded on each individual channel

$$y_i(j) = s_i(j) - \frac{1}{N}\sum_{l=1}^{N} s_l(j) \quad (1)$$

where $s_i(j)$ denotes the signal recorded on channel i at time j, $y_i(j)$ is the filtered signal after subtracting the mean of all channels and N is the number of channels [9,11]. We then applied a band-bass filter to y_i in the range of 0.1–10 Hz.

We then extract the time segment that could contain the P300 potential from the recorded EEG signals. Typically, the P300 potential appears in the range 250–500 ms so the minimum time window to contain P300 potential should be much larger than 500 ms [3]. Across many studies, the used time window ranges from 600 ms to 1000 ms [11–15]. As a result, we use a window of 90 samples which corresponds to a time window of 700 ms based on our previous analysis as mentioned earlier [9]. Finally, after extracting the epoch, z-score is then applied to each channel epoch

$$x_{ij} = (y_i(j) - \mu_i)/\sigma_i \quad (2)$$

where x_{ij} represents normalized sample j of channel i in the current epoch and μ_i and σ_i are the mean and standard deviation of channel i, respectively.

2.3. Trial averaging and variable trial schemes

After preprocessing the recorded signals, the extracted epochs of all trials for each character are averaged to reduce the noise. As mentioned earlier, we used 10 trials for each character. In the training phase, all 10 trials are averaged to form the final epoch features that are used in classification. However, using a large number of trials comes at the expense of the speed of spelling. This is not a big issue in training as it is done once, however, in testing, it is crucial to make testing as fast as possible while maintaining high accuracy. Therefore, in our experiments, we compared the results of different methods when using all 10 trials in testing and when using variable number of trials (referred to as variable trial scheme hereafter). In the variable trial scheme, we start by classifying the first trial to identify the detected character. In each subsequent trial, we average the trial with all preceding trials and classify the average to identify the detected character. If two successive classifications result in the same character, we stop. Thus, the minimum number of trials needed for the method to detect a character is two. This approach is consistent with the adaptive approaches utilized in [16,17].

2.4. Feature extraction and classification

P300 data comprises both spatial and temporal features. Typically, the corresponding feature vector is formed by extracting features in a specific time window from all channels then concatenating all these features into one feature vector. Given the large number of dimensions in this case, dimensionality reduction methods are typically utilized to identify the most relevant features from the data. One common dimensionality reduction method is Principal Component Analysis (PCA) which is a statistical tool that uses eigenvalue decomposition of data covariance matrix to identify the dimensions, called principal components, that best represent the input data [18]. This is done by, first, subtracting the mean of each feature to center the data. The covariance matrix is then computed based on the centered data. The eigenvectors of the covariance matrix are then obtained and sorted based on their corresponding eigenvalues. PCA has been successfully used in P300 applications as reported in [11,15,19]. However, given the high computational complexity of PCA when used with large feature vectors as in the EEG analysis case, it might not be suitable to use on low computational-power devices, such as mobile devices, without performing additional pre-processing to reduce the dimensionality of the input data.

Another dimensionality reduction method that is typically used is

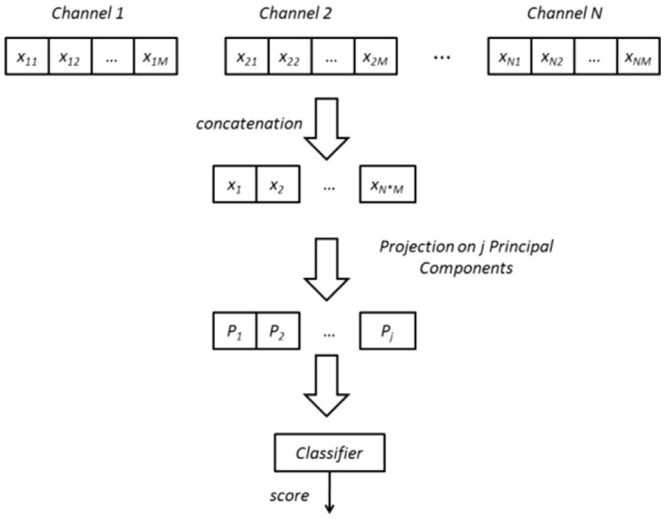

Fig. 4. Concatenated feature vector for traditional approaches where the samples of all channels are combined in one feature vector. This feature vector is then projected on the principal components to obtain the reduced dimensions feature vector that represents the input to a classifier.

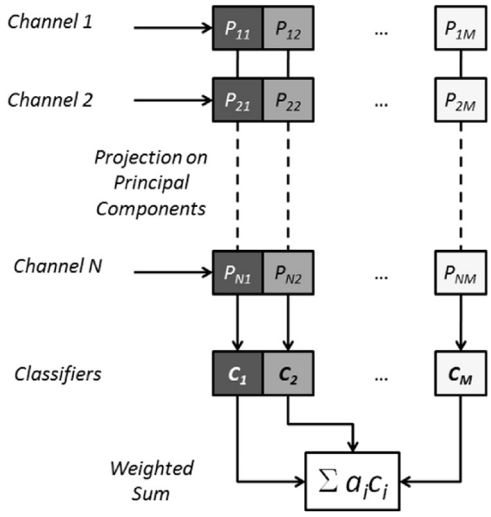

Fig. 5. PCA Ensemble Classifier Approach. Signals recorded on each channel are projected using the corresponding principal components. Projections of the *i*th principal component of all channels are concatenated to form a feature vector. A classifier is the used to classify the feature vector of each principal component. The final score is a weighted summation of the ensemble classifier scores.

decimation which is considered a very simple, yet effective, method that discards some samples from the feature vector (i.e. downsampling the signal). However, decimation should not be used solely as it does not select the most relevant features, but only reduces them efficiently [11,20]. For instance, as the number of recorded channels used in the analysis increases, a larger decimation factor is required to maintain a fixed number of features compared to using a smaller number of channels. This in turn reduces the quality of the recorded signals which could reduce the overall performance of the system [20]. As a result, a channel selection algorithm is typically used with decimation to select the most relevant channels [21].

As a result, a hybrid approach in which both PCA and decimation are used could result in better performance in which the time segments obtained from all channels for one character epoch are concatenated to constitute one feature vector as illustrated in Fig. 4 which represents the traditional feature extraction approach. We also investigated the performance in the case when PCA is only used without decimation. The number of principal components taken is based on a variance threshold of 0.9995 which is chosen based on our previous results [9].

To determine the target character from one set of intensifications, the concatenated feature vector corresponding to the intensification of each row and each column is classified. In case more than one row or column are classified as the target choice, the target row and column are determined as those that maximize the classifier score where

$$predictdedrow = \arg\max_{row}(score_{row}) \tag{3}$$

$$predictdedcol = \arg\max_{col}(score_{col}) \tag{4}$$

2.5. PCA ensemble classifier

We introduce a feature extraction method that is used with an ensemble classifier to efficiently discriminate among the two signals: P300 versus non-P300. The intuition is that we divide the features (i.e. variables) over multiple classifiers while keeping the same data size for each classifier which makes each classifier extract better features. In this approach, PCA is first applied to the recorded preprocessed EEG signal of each channel *individually* to identify M principal components. These principal components are sorted based on the variance in the data accounted for by each principal component. We select the principal components threshold by, first, computing the average of the variance accounted for by each principal component across channels and then selecting the principal components that account for 0.9995 of variance. The projections of the input signals on the *i*th principal component of all channels are then combined and used to train M classifiers where a classifier is constructed for each extracted principal component as illustrated in Fig. 5. When applying the trained ensemble classifier to test data, we use a weighted summation scheme in which the output of each classifier is multiplied by a weight that represents the significance of the corresponding principal component.

In the proposed method, we use an ensemble classifier in which a classifier for each principal component is trained and the total score is a weighted sum of the classifiers scores

$$totalscore = \sum_{i=1}^{M} (a_i * score_i) \tag{5}$$

where M is the number of classifiers and $score_i$ is the score of the *i*th classifier. Different weight functions examined in our analysis are detailed in the next section.

2.6. PCA ensemble classifier weights

The score obtained from each classifier has to be multiplied by a weight that reflects the significance of the corresponding principal component. We propose three weight measures based on the eigenvalues and examine the performance of each one in cross-validation and offline testing.

First, we average the corresponding eigenvalues from all channels since all examined weight measures are based on these values as

$$\lambda_j = \sum_{k=1}^{N} \lambda_{jk}/N \tag{6}$$

where λ_{jk} represents eigenvalue j of channel k obtained from PCA analysis and N denotes the number of channels. For all weight measures, the first principal component is the most significant and we denote it by index 1. We examined the following weights:

1) Inverse of Accumulated Eigenvalue

This measure assigns to the score obtained from classifier i the inverse of accumulated eigenvalues defined in (6) from 1 to i

$$a_i = 1/\left(\sum_{j=1}^{i} \lambda_j\right) \tag{7}$$

As a result, significant principal components will be assigned relatively large weights compared to insignificant principal components. The weights of this measure decay slowly compared to the following weights such that the difference in the weights of successive principal components is relatively small.

2) Eigenvalue

This measure assigns to principal component classifier i the average eigenvalue of all ith principal components of all channels as calculated in (6). This measure gives more weight to significant components while weights are changing fast for the first principal component

$$a_i = \lambda_i \tag{8}$$

3) Square Root of Eigenvalue

This measure assigns to principal component classifier i the square root of the average eigenvalue of all ith principal components of all channels as calculated in (6). We use the square root to limit the changing rate of the weights.

$$a_i = \sqrt{\lambda_i} \tag{9}$$

Finally, similar to the concatenated feature vector case, the predicted row and column for the target character can be obtained using Eqs. (3) and (4).

2.7. Classification methods

P300 classification is a binary classification problem where the classifier discriminates between the P300 class and the non-P300 class. Previous studies have shown that both linear and non-linear classifiers can be used in the P300 classification problem with acceptable results [11,14,22]. However, the advantage of linear classifiers is that they are simple and computationally less expensive compared to non-linear classifiers which makes linear classifiers more suitable for mobile devices. The decision boundary of any linear classifier takes the form

$$w^T x + b = 0 \tag{10}$$

where w denotes the weight vector of the classifier, x is the feature vector obtained using the aforementioned feature extraction method, and b is the bias term. A classifier is trained by calculating the weight vector and the bias term from the training data so that the classifier can classify any new unlabeled feature vector. We examined the following classifiers in our analysis:

1) *Least Squares (LS)* [23] is a linear classifier that minimizes the squared error between the target label and the decision boundary score. The solution of this minimization takes the form

$$w = (X^T X)^{-1} X^T t \tag{11}$$

where X is the data matrix whose rows are the extracted feature vectors and t is the labels vector which includes the target label (whether P300 or non-P300) of each feature vector. LS has been previously used in [11] where it achieved ~90% classification accuracy using a traditional EEG recording devices and a desktop P300-paradigm application.

2) *Fisher's Linear Discriminant (FLD)* [23] is used to find the separating decision boundary between the two classes that maximizes the separation between the two classes while minimizing the variance of each class. The weight vector of FLD is given by

$$w = S_W^{-1}(m_1 - m_2) \tag{12}$$

where m_1 and m_2 denote the means for each of the P300 and non-P300 classes while S_W is the within-class scatter which is given by

$$S_W = \mathrm{cov}(X_{P300}) + \mathrm{cov}(X_{non-P300}) \tag{13}$$

where cov(.) denotes the covariance, X_{P300} is a matrix with rows representing P300 feature vectors while $X_{non\text{-}P300}$ is a matrix with rows representing non-P300 feature vectors. FLD has been used in multiple P300 data studies where it achieved a classification accuracy of ~90% in [11], ~100% in [19], ~80% in [24] when it was co-trained with Bayesian Linear Discriminant Analysis classifier, ~80% in [14], and ~85% in [15]. All these results were obtained using traditional EEG recording devices and desktop P300-paradigm applications.

2.8. Bitrate calculation

To best compare methods when using variable number of trials, we calculated the bitrate achieved using each method. The number of bits used in bitrate calculation is calculated as [25]

$$B = \log_2(N) + p\log_2(p) + (1-p)\log_2\left(\frac{1-p}{N-1}\right) \tag{14}$$

where B is the average number of bits, N is the number of symbols, and p is the accuracy. Given that we have a 6×6 grid, the number of symbols N =36 symbols. The bitrate per trial b_{TR} is then computed as

$$b_{TR} = \frac{B}{TR_{avg}} \tag{15}$$

where TR_{avg} is the average number of trials. The bitrate per minute b_{min} is given by

$$b_{\mathrm{min}} = \frac{B}{T_{avg}} \tag{16}$$

where T_{avg} is the average symbol selection time.

3. Results

3.1. P300 potentials

We first demonstrate samples of the P300 potentials obtained using the Emotiv EPOC neuroheadset from the recorded data on Android platforms. The more discriminable the P300 potential from ongoing activity signals, the higher the classification accuracy we could achieve. Fig. 6a and b demonstrate the feasibility of obtaining distinct P300 potentials using such setup.

3.2. Cross validation

We performed cross validation on the data recorded on Android from six subjects using an inter-stimulus interval (ISI) of 300 ms, a fixed time window of 90 samples, and PCA threshold of 0.9995 [9]. We used a training data size of 33 characters, where the number of trials for each character epoch was 10 trials. We divided the 33 characters training data into 25 for training and 8 characters for validation. We formed 26 validation subsets each of 8 characters with an overlap of 7 characters. The average cross validation result of each feature extraction and classification method of all six subjects is illustrated in Fig. 7a in which the accuracy was computed as the percentage of correctly classified characters. A maximum accuracy of 93.75 ± 8.1% was achieved with maximum mean classification accuracy across all subjects of 78.37 ± 16.09% obtained for the least squares ensemble classifier with inverse of accumulated eigenvalues weights defined in Eq. (7). For the traditional single classifier, the highest mean accuracy achieved was 70.83 ± 22.73% using least squares method. The results demonstrate that, despite using limited training data of only 33 characters and 10 trials/character, a relatively high classification accuracy can still be achieved. Our proposed ensemble classifier

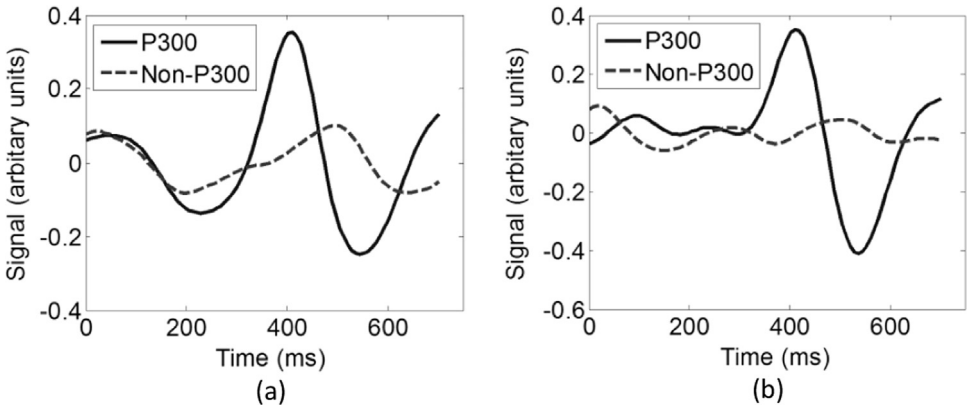

Fig. 6. Average P300 and non-P300 potentials evoked on channels (a) F8 and (b) AF4 for one subject (Subject S1). The figure indicates a significant difference between P300 and non-p300 potentials.

outperformed the traditional concatenated method that uses a single classifier. In addition, the results suggest that using the inverse weight measure achieves a higher accuracy compared to other weight measures.

To investigate how the proposed weight measures weigh each principal component, we examined the mean normalized weights across all subjects for the first 13 principal components. As observed in Fig. 7b, our analysis reveals that the inverse accumulated weight measure tends to have slightly changing weights between principal components while eigenvalues weight measure tends to have the largest changes between weights. This suggests that a property of the best weight measure is to not decrease significantly for higher principal component numbers.

3.3. Offline testing

We recorded testing data online using *MindEdit* interface from all six subjects. The results of offline test of this data for all methods versus the number of trials used are shown in Fig. 8a for FLD classifier and Fig. 8b for LS classifier. Results demonstrate that the proposed ensemble classifier method with inverse accumulated weights is the best classifier for the FLD classifier for all number of trials and for the LS classifier in most of them. This is consistent with the results obtained from the cross validation analysis. The proposed ensemble classifier with eigenvalues square root weights comes next. Using all 10 trials, our proposed method with inverse accumulated weights achieved mean accuracy of 77.5 ± 19.69% and 74.17 ± 19.08% across all subjects with LS and FLD methods, respectively. The proposed method with square root weights achieved 75 ± 14.49% and 71.67 ± 16.93%, respectively. Finally, the traditional method achieved 71.67 ± 10.8% and 67.5 ± 12.55%, respectively. As a result, the proposed methods are capable of extracting better features compared to the traditional concatenated method.

3.4. Variable trial offline testing

We further extended our analysis to use variable number of trials instead of fixed number of trials to best compare all methods. From Fig. 9a and b, the proposed classifier with inverse accumulated weights achieved 41.6 ± 12.11% and 42.5 ± 12.94% using LS and FLD, respectively. The number of trials taken to achieve such accuracies was 3.53 ± 1.48 and 3.42 ± 1.37, respectively. The traditional method achieved 44.17 ± 7.36% and 41.67 ± 5.16% using LS and FLD, respectively, taking 3.67 ± 1.58 and 3.72 ± 1.63 trials, respectively, which are more than what is needed for the proposed method to converge. These results could be interpreted better by calculating the bitrate as will be shown later.

3.5. Variable trial online testing

We performed an online testing of the proposed method using FLD classifier and inverse accumulated weights, and the traditional methods to compare their performance in real time on the Android platform. All subjects attempted to spell the 20-character sentence “AIN SHAMS UNIVERSITY”. We did the online test for an ISI of 300 ms. Results illustrated in Fig. 10 show that our approach achieved mean classification accuracy of 64.17 ± 19.6% compared to 42.5 ± 11.29% achieved using the traditional method in 3.41 ± 1.51 trials compared to 3.39 ±

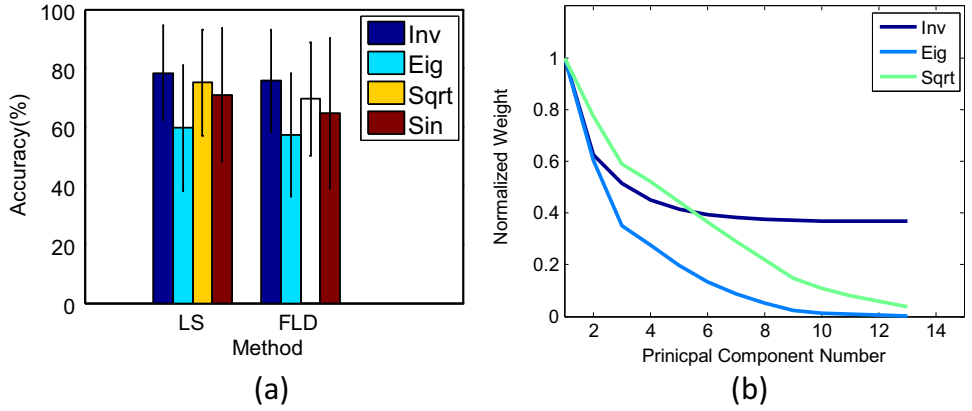

Fig. 7. (a) Cross validation mean accuracy for different methods (mean ± s.d.), where Inv corresponds to the Inverse of Accumulated Eigenvalue weight, Eig corresponds to using the eigenvalue itself as a weight, Sqrt corresponds to the Square Root of Eigenvalue weight, while Sin corresponds to the traditional concatenated method. (b) Value of different weights versus principal component number.

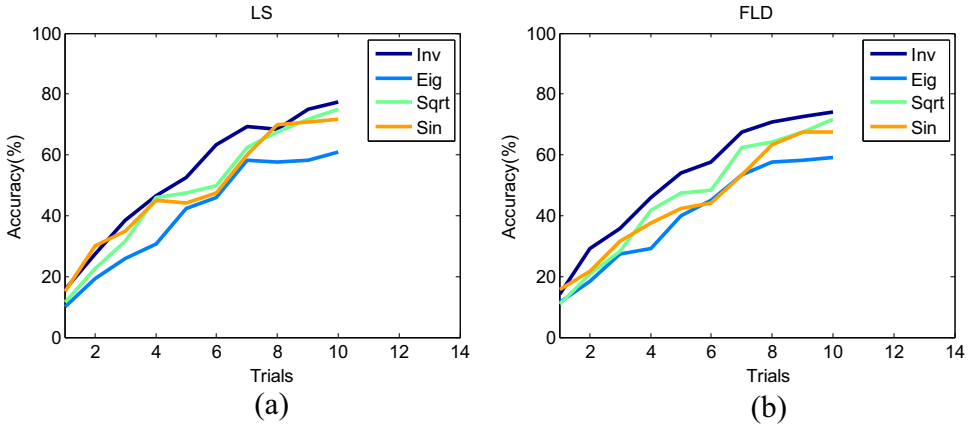

Fig. 8. Offline average classification accuracy across subjects versus number of trials using (a) LS and (b) FLD classifiers.

1.58 for the traditional method. The best classification accuracy achieved using the proposed method was 85% while for the traditional method was 60%.

3.6. Bitrate results

Using the Eqs. (14)–(16), we calculated the bitrate for all methods for all testing approaches performed in previous sections: offline testing with all 10 trials, offline testing with variable trials and online testing with variable trials. The results are summarized in Tables 2, 3, and 4, respectively. The tables demonstrate that the proposed method has higher bitrates than the traditional method when using the proposed ensemble classifier with the inverse weight measure and Fisher's linear discriminant classifier. More importantly, the proposed method achieved a maximum bitrate of 6.25 bit/min compared to 3.1 bit/min for the traditional method in the case of online variable trials test.

3.7. Variable trial scheme analysis

The main purpose of the variable trial scheme is to enhance the bit rate of *MindEdit* compared to using all 10 trials. In the used scheme, we compare the classification result obtained using the first trial to the result obtained using the average of the first and second trials. If the same character is recognized in both cases, the algorithm stops and this character is considered as the character detected by the algorithm. If not, we compare the classification result obtained using the average of the first two trials to that obtained using the average of the first three trials. This process continues by adding one trial to the average until the classification result obtained using two successive averages is the same (i.e. same character is recognized in both successive averages). To justify using only two successive averages for convergence, we first computed the probability of getting the same classification result from two successive averages for the offline testing data using the proposed ensemble classifier with the inverse weight measure. We compared this probability to using higher number of successive averages as shown in Table 5. The table demonstrates that achieving similar classification results from two successive averages is the most likely case. We then computed the bitrates that would be achieved using different number of successive averages for convergence as summarized in Table 6. The results show that using two successive averages achieves bitrate of 3.1 bit/min which is slightly lower than the highest bitrate achieved using three successive averages (3.3 bit/min). As a result, we used two successive averages in our analysis given that achieving similar classification result using two successive averages occurs with a larger probability compared to using three successive averages as shown in Table 5 and given the slight difference in bitrates. In addition, in this analysis, feature extraction, averaging, and classification times were not accounted for in bitrate calculation of the variable trial scheme. Taking them into account would penalize the use of higher number of successive averages. This also confirms that using two successive averages is more efficient which is consistent with previous studies of adaptive trial schemes [16,17].

3.8. Complexity analysis

To examine the computational efficiency of the PCA ensemble classifier compared to the traditional concatenated feature vector method, we performed a timing analysis of both approach. Computational efficiency is crucial given the limited resources typically available on mobile devices. First, assume that we have N channels each with M samples and n number of training points. The features

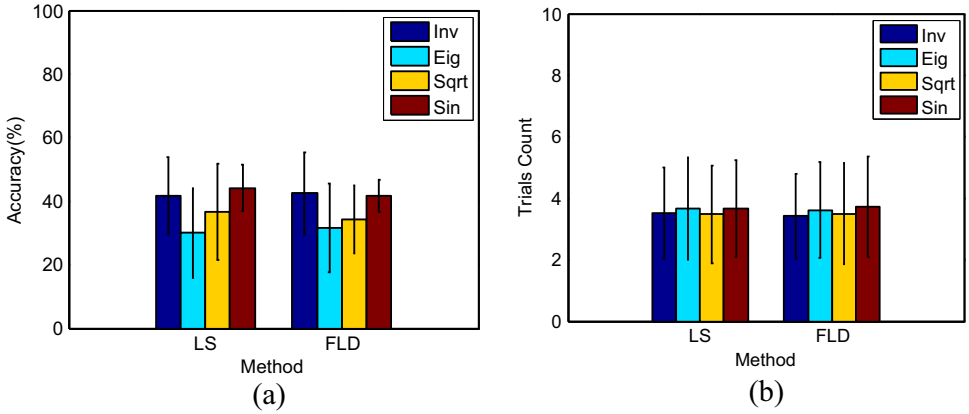

Fig. 9. (a) Mean offline classification accuracy using variable trials. (b) Mean number of trials in this case.

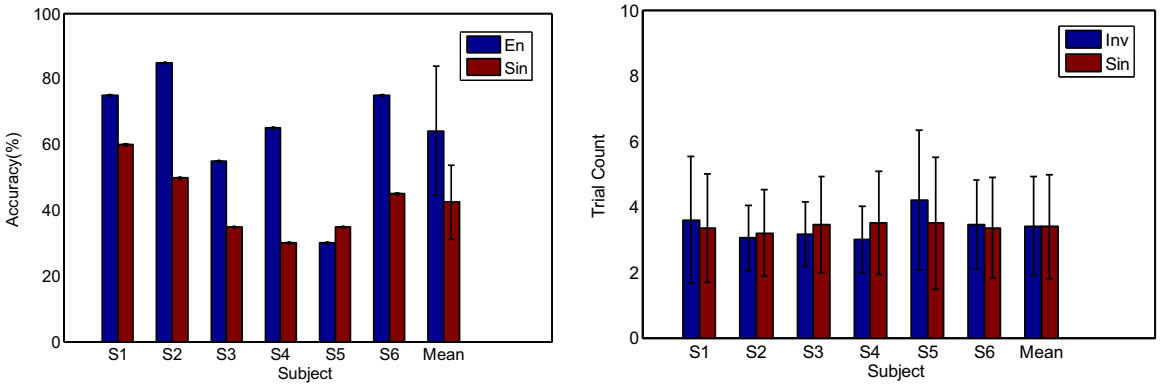

Fig. 10. (a) Online classification accuracy using variable trials for each subject. (b) Mean number of trials in this case.

Table 2
Bitrate for all methods using all trials for offline data.

Method	Bitrate (Bit/trial)		Bitrate (Bit/min)	
	LS	FLD	LS	FLD
Inv	0.34	0.31	2.78	2.57
Eig	0.27	0.22	1.87	1.81
Sqrt	0.31	0.29	2.59	2.42
Sin	0.29	0.26	2.38	2.17

Table 3
Bitrate for all methods using variable number of trials for offline data.

Method	Bitrate (Bit/trial)		Bitrate (Bit/min)	
	LS	FLD	LS	FLD
Inv	0.38	0.37	2.94	3.07
Eig	0.21	0.23	1.74	1.88
Sqrt	0.3	0.26	2.45	2.17
Sin	0.37	0.33	3.05	2.69

Table 4
Bitrate for variable number of trials for online data.

Method	Bitrate (Bit/trial)	Bitrate (Bit/min)
Inv (FLD)	0.76	6.25
Sin (FLD)	0.38	3.1

Table 5
Occurrence probabilities of achieving similar classification using different number of successive averages.

Successive Averages Number	2	3	4	5	6	7	8	9	10
Occurrence Probability	0.65	0.5	0.4	0.33	0.28	0.22	0.17	0.11	0.05

Table 6
Bitrate obtained using different number of successive averages.

Successive Averages Number	2	3	4	5	6	7	8	9	10
Bitrate (bit/min)	3.1	3.3	3.1	3	2.8	2.7	2.6	2.6	2.6

matrix X for the traditional concatenated method will be of dimensions n x NM while in the PCA ensemble classifier method, we have a features matrix for each channel (i.e. N feature matrix) of dimensions n x M. Table 7 summarizes the PCA steps and the complexity of each step

Table 7
Complexity of each step of PCA [26,27].

Step	Complexity of Concatenated Method	Complexity of Proposed Method	
		Single Channel	N Channels
Mean	$O(nNM)$	$O(nM)$	$O(nNM)$
Centering	$O(nNM)$	$O(nM)$	$O(nNM)$
Covariance	$O(nN^2M^2)$	$O(nM^2)$	$O(nNM^2)$
Eigen Decomposition	$O(N^3M^3)$	$O(M^3)$	$O(NM^3)$
Projection	$O(nNMP)$	$O(nMR)$	$O(nNMR)$

and how it is reduced in our proposed method. If the number of components after projection in the concatenated traditional method is P and in the proposed methods is R for each channel, the mean and centering operations are in order of the matrix size (i.e. $O(size)$). The covariance matrix is the product X^TX, where X is the feature matrix and X^T is the transpose of X, so the operation is in order of the size of the left matrix times the number of columns of the right matrix (i.e. $O(size1 \times col2)$). The dimensions of covariance matrices are NM x NM for the traditional concatenated feature vector method and M x M for the proposed method. The eigenvalue decomposition is in the order of square matrix product so it is in the order of the cube of the matrix rows (i.e. the number of rows or columns as it is square matrix, $O(rows^3)$). Finally, the projection is just a multiplication operation [26,27].

The way we construct feature vectors is based on principal components grouping which results in M final feature matrices at maximum, each of dimensions n x N. This has the effect of reducing the complexity of the classifier training computations. Table 8 illustrates the complexity of operations used in LS and FLD classifiers. The heaviest operations are the calculations of the covariance matrix and the matrix inversion.

Table 8
Complexity of heaviest operations in classifiers.

Operation	Complexity of Concatenated Method	Complexity of Proposed Method	
		Vector	M Vectors
Covariance	$O(nN^2M^2)$	$O(nN^2)$	$O(nN^2M)$
Inversion	$O(N^3M^3)$	$O(N^3)$	$O(N^3M)$

4. Discussion and conclusion

MindEdit was developed to advance the practical application of BCIs one further step in which the use of P300-based BCIs is manifested in interacting with mobile devices such as tablets and smartphones. To the best of our knowledge, this application is the first demonstration of a BCI application that enables typing characters using brain activity on mobile devices. Developing such application and similar ones could improve the quality of life for people with hand disability who are not able to interact with touchscreen-based devices. Moreover, it can be beneficial for healthy subjects who cannot type during some daily life activities such as driving or exercising.

The offline testing results reported here using the proposed PCA ensemble classifier applied to the average of 10 trials are not inferior to previous reports [11,28–30]. It is important to note that such results were obtained despite the use of the Emotiv wireless headset and a tablet screen for the stimulation paradigm as opposed to using traditional EEG recording systems and larger-size computer screens. Compared to other P300-based mobile BCI applications that we and others have previously developed [31,32], the performance reported here is comparable given that the grid used in *MindEdit* is of size 6×6 which typically results in a diminished performance compared to using smaller grid sizes [33]. In addition, our results indicated that the proposed feature extraction method for the PCA ensemble classifier (using different weight functions) outperforms the use of traditional concatenated feature vectors. In addition, using the proposed inverse weight measure outperforms other suggested weight measures.

Given that *MindEdit* is developed as a mobile application, maintaining a relatively low computational complexity represents a central aspect of the design. The proposed ensemble classifier was designed such that it possesses lower computational complexity compared to the traditional feature vector approach. When using devices with limited resources such as mobile devices, using algorithms such as step-wise linear discriminant analysis (SWLDA) would not be suitable given the relatively high complexity [34]. For instance, SWLDA preforms a search to identify significant features and discard insignificant features which contributes heavily to its computational complexity compared to the proposed approach. In addition, the main motivation of the proposed approach is to identify the most relevant features from each channel and combine such features. This results in lower computational complexity compared to other methods that analyze data from *all* channels simultaneously.

In *MindEdit*, we have adopted a variable number of trials scheme as opposed to using the average of 10 trials to reach a decision. Such scheme had an impact on the bitrate of the system increasing it from 2.57 bit/min in the case of using the average of 10 trials to 3.07 bit/min when using variable number of trials scheme. In the online testing, using the variable number of trials resulted in a more prominent improvement in the bitrate reaching 6.25 bit/min. The introduced variable trial scheme can circumvent the slow operation typically associated with P300-based BCIs by acting as a hybrid approach of fast but inaccurate single-trial and slow but more accurate multiple-trial schemes [35,36]. Contrary to other studies that indicated that similar adaptive variable trial schemes could achieve better accuracy compared to the traditional multiple trial schemes [16], the increase in bitrate was achieved with a slight reduction in accuracy. Using *MindEdit*, an online accuracy of 64.17 ± 19.6% was obtained when using the variable number of trials scheme compared to an offline accuracy of 77.5 ± 19.69% obtained using the 10-trial average scheme. However, the elevated bitrate achieved using this scheme could compensate for the slight reduction in accuracy where an error could be corrected by selecting the 'Backspace' symbol.

Our analysis revealed that the accuracy obtained in the online testing is much higher than that obtained in the offline testing. Such result is consistent with some previous reports that compared online to offline analysis [16,31,33,37]. The elevated online accuracy could be attributed to the reduction in the number of needed trials in the online operation of *MindEdit* compared to the offline analysis in which the analyzed data was originally a 10-trial training dataset. In addition, while recording training data for offline analysis, subjects are not typically fully engaged in the task as they are directed to focus on a specific character whereas in the online operation, subjects are more engaged as the application is completely driven by the subject's recorded activity.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of interest statement

None declared.

References

[1] B. Graimann, B. Allison, G. Pfurtscheller, Brain–Computer Interfaces: A Gentle Introduction, Brain-Computer Interfaces, Springer, Berlin Heidelberg, 2010, pp. 1–27.
[2] L.A. Farwell, E. Donchin, Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials, Electroencephalogr. Clin. Neurophysiol. 70 (1988) 510–523.
[3] J. Polich, Updating P300: an integrative theory of P3a and P3b, Clin. Neurophysiol. 118 (2007) 2128–2148.
[4] A.Campbell, T.Choudhury, S.Hu, H.Lu, M.K.Mukerjee, M.Rabbi, R.D.Raizada, NeuroPhone: brain-mobile phone interface using a wireless EEG headset, in: Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds, ACM, New Delhi, India, August 30, 2010, pp. 3–8
[5] A. Stopczynski, C. Stahlhut, J.E. Larsen, M.K. Petersen, L.K. Hansen, The smartphone brain scanner: a portable real-time neuroimaging system, PloS One 9 (2014) e86733.
[6] Y.-T. Wang, Y. Wang, T.-P. Jung, A cell-phone-based brain–computer interface for communication in daily life, J. Neural Eng. 8 (2011) 025018.
[7] M. Caruso, F. Cincotti, F. Leotta, M. Mecella, A. Riccio, F. Schettini, L. Simione, T. Catarci, My-World-in-My-Tablet: An Architecture for People with Physical Impairment, in: M. Kurosu (Ed.)Human-Computer Interaction. Interaction Modalities and Techniques,, Springer, Berlin Heidelberg, 2013, pp. 637–647.
[8] J.E. Council, Epilepsy Prevalence, Incidence and Other Statistics, Joint Epilepsy Council, Leeds, UK, 2005.
[9] A.S.Elsawy, S.Eldawlatly, M.Taher, G.M.Aly, Performance analysis of a Principal Component Analysis ensemble classifier for Emotiv headset P300 spellers, Engineering in Medicine and Biology Society (EMBC), 2014 in: Proceedings of the 36th Annual International Conference of the IEEE, IEEE, 2014, pp. 5032–5035
[10] D.J. McFarland, L.M. McCane, S.V. David, J.R. Wolpaw, Spatial filter selection for EEG-based communication, Electroencephalogr. Clin. Neurophysiol. 103 (1997) 386–394.
[11] D.J. Krusienski, E.W. Sellers, F. Cabestaing, S. Bayoudh, D.J. McFarland, T.M. Vaughan, J.R. Wolpaw, A comparison of classification techniques for the P300 speller, J. Neural Eng. 3 (2006) 299.
[12] M.Kaper, P.Meinicke, U.Grossekathoefer, T.Lingner, H.Ritter, BCI competition 2003-data set IIb: support vector machines for the P300 speller paradigm, IEEE Trans. Biomed. Eng., 51, 1073–1076, 2004.
[13] A. Rakotomamonjy, V. Guigue, BCI competition III: dataset II-ensemble of SVMs for BCI P300 speller, IEEE Trans. Biomed. Eng. 55 (2008) 1147–1154.
[14] N.V. Manyakov, N. Chumerin, A. Combaz, M.M. Van Hulle, Comparison of classification methods for P300 brain-computer interface on disabled subjects, Comput. Intell. Neurosci. 2011 (2011) 2.
[15] A. Finke, A. Lenhardt, H. Ritter, The MindGame: a P300-based brain–computer interface game, Neural Netw. 22 (2009) 1329–1333.
[16] J. Jin, B.Z. Allison, E.W. Sellers, C. Brunner, P. Horki, X. Wang, C. Neuper, An adaptive P300-based control system, J. Neural Eng. 8 (2011) 036006.
[17] J. Jin, B.Z. Allison, T. Kaufmann, A. Kübler, Y. Zhang, X. Wang, A. Cichocki, The changing face of P300 BCIs: a comparison of stimulus changes in a P300 BCI involving faces, emotion, and movement, PLoS One 7 (2012) e49688.
[18] H. Abdi, L.J. Williams, Principal component analysis, Wiley Interdiscip. Rev.: Computat. Stat. 2 (2010) 433–459.
[19] H.Mirghasemi, R.Fazel-Rezai, M.Shamsollahi, Analysis of P300 classifiers in brain computer interface speller, in: Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS'06). IEEE, New York City, New York, August 31- September 3, 2006, pp. 6205–6208
[20] D.J. Krusienski, E.W. Sellers, D.J. McFarland, T.M. Vaughan, J.R. Wolpaw, Toward enhanced P300 speller performance, J. Neurosci. Methods 167 (2008) 15–21.
[21] E. Yin, T. Zeyl, R. Saab, D. Hu, Z. Zhou, T. Chau, An auditory-tactile visual saccade-independent P300 brain–computer interface, Int. J. Neural Syst. 26 (2016) 1650001.

[22] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, B. Arnaldi, A review of classification algorithms for EEG-based brain–computer interfaces, J. Neural Eng. 4 (2007).
[23] R.O.Duda, P.E.Hart, D.G.Stork, Pattern classification, John Wiley & Sons, 2012.
[24] R.C. Panicker, S. Puthusserypady, Y. Sun, Adaptation in P300 brain–computer interfaces: a two-classifier cotraining approach, IEEE Trans. Biomed. Eng. 57 (2010) 2927–2935.
[25] D.J. McFarland, W.A. Sarnacki, J.R. Wolpaw, Brain–computer interface (BCI) operation: optimizing information transfer rates, Biol. Psychol. 63 (2003) 237–251.
[26] P.Brgisser, M.Clausen, M.A.Shokrollahi, Algebraic Complexity Theory, Springer Publishing Company, Incorporated, 2010.
[27] H. Lu, K.N. Plataniotis, A. Venetsanopoulos, Multilinear Subspace Learning: Dimensionality Reduction of Multidimensional Data, CRC Press, Boca Raton, FL, USA, 2013.
[28] J. Lu, W. Speier, X. Hu, N. Pouratian, The effects of stimulus timing features on P300 speller performance, Clin. Neurophysiol. 124 (2013) 306–314.
[29] A. Kübler, N. Birbaumer, Brain–computer interfaces and communication in paralysis: extinction of goal directed thinking in completely paralysed patients?, Clin. Neurophysiol. 119 (2008) 2658–2666.
[30] M. Salvaris, F. Sepulveda, Visual modifications on the P300 speller BCI paradigm, J. Neural Eng. 6 (2009) 046011.
[31] A.S.Elsawy, S.Eldawlatly, P300-based applications for interacting with smart mobile devices, 2015 in: Proceedings of the 7th International IEEE/EMBS Conference on Neural Engineering (NER), IEEE, 2015, pp. 166–169
[32] T.Jijun, Z.Peng, X.Ran, D.Lei, The portable P300 dialing system based on tablet and Emotiv Epoc headset, 2015 in: Proceedings of the 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE, 2015, pp. 566–569
[33] E.W. Sellers, D.J. Krusienski, D.J. McFarland, T.M. Vaughan, J.R. Wolpaw, A P300 event-related potential brain–computer interface (BCI): the effects of matrix size and inter stimulus interval on performance, Biol. Psychol. 73 (2006) 242–252.
[34] E.W. Sellers, E. Donchin, A P300-based brain–computer interface: initial tests by ALS patients, Clin. Neurophysiol. 117 (2006) 538–548.
[35] B. Blankertz, S. Lemm, M. Treder, S. Haufe, K.-R. Müller, Single-trial analysis and classification of ERP components—a tutorial, NeuroImage 56 (2011) 814–825.
[36] I.P. Ganin, S.L. Shishkin, A.Y. Kaplan, A. P300-based, brain-computer interface with stimuli on moving objects: four-session single-trial and triple-trial tests with a game-like task design, PloS One 8 (2013) e77755.
[37] E. Yin, Z. Zhou, J. Jiang, F. Chen, Y. Liu, D. Hu, A speedy hybrid BCI spelling approach combining P300 and SSVEP, IEEE Trans. Biomed. Eng. 61 (2014) 473–483.