A Closed-loop Adaptive Brain-computer Interface Framework: Improving the Classifier with the Use of Error-related Potentials

Kuan-Jung Chiang, Dimitra Emmanouilidou, Hannes Gamper, David Johnston,
Mihai Jalobeanu, Edward Cutrell, Andrew Wilson, Winko W. An., and Ivan Tashev

***Abstract*— Brain-computer interfaces (BCIs) using Electroencephalography (EEG) have drawn attention to providing alternative control pathways for users with motor disabilities or even the general public in real-world environments due to their robustness, relatively low cost, and high portability. However, EEG still suffers from large variability between subjects or between sessions of an individual subject. To obtain optimal performance, a BCI usually requires a user to go through a calibration process to fine-tune the model. This calibration process is usually long and could hinder the practicality of a BCI. In this study, we propose a closed-loop framework that monitors the user EEG responses to the action of a BCI. If an Error-related Potential (ErrP) is detected in the response, it is indicated that the BCI is making a wrong prediction. By using the information from this ErrP detector, we can include online testing trials into the training pool and further fine-tune the model over the time the BCI is used. Results suggest that the proposed framework can reach better results with a few additional trials when compared to the model pre-trained from some existing data. Also, the performance of the proposed model can gradually converge to a fully calibrated model, which suggests that the conventional calibration process could be replaced by online training.**

I. INTRODUCTION

A brain-computer interface (BCI) provides a pathway for users to control computers or machines with their brain activities. Among various types of BCI modalities, electroencephalography (EEG) which measures voltage fluctuations resulting from ionic current within the neurons of the brain has been the most popular one due to its high temporal resolution, high portability, and relative straightforward set-up process [1], [2]. Real-world applications for EEG-based BCIs have been proposed previously. For example, P300-based spellers [3] and SSVEP-based spellers [2] were built for people with motor disability. Also, VR headsets combined with EEG sensors were proposed to enhance user interfaces in VR games [4]. These examples demonstrate the feasibility of EEG-based BCIs in real life.

However, EEG-based BCIs are still impractical for many application scenarios. One of the major challenges is that to obtain a robust performance in decoding the EEG signals, a calibration process is usually required to optimize the statistical model because there is a large subject-to-subject variability or relatively small session-to-session variability within an individual. This calibration process is usually time-consuming which hinders the practicality of the BCIs.

In order to reduce the calibration time, transfer learning approaches have been proposed to leverage existing data from other users or data from previous sessions of the same user [5]. Studies have shown that a BCI decoder can start with a model trained with some existing data, and adaptive learning methods can be used to progressively fine-tune the model during the time the system is used [6]–[8]. In this work, we propose a closed-loop adaptive BCI framework that consists of two main components, a control classifier, and an Error-related Potential (ErrP) detector. The control classifier decodes the EEG signals of the user's intention to send commands as a regular BCI, while the ErrP detector monitors the user's EEG responses to the result the control classifier outputs. We show that in this framework, a BCI system is able to collect new training trials with their pseudo-labels, and the performance of the control classifier improves over the time the system is used.

II. METHOD

A. Framework

Our proposed framework consists of two main components:

1) The SSVEP classifier: It serves as the control classifier in a traditional BCI system which decodes user SSVEP responses to the stimuli and translates them into commands.
2) The ErrP detector: It is an additional component compared to a traditional BCI system. This detector decodes the EEG responses shortly after the output of the SSVEP classifier is displayed to the user to detect whether an ErrP exists.

We chose to use SSVEP as the BCI paradigm in our framework because SSVEP decoding is relatively well developed and studied. Thus, we could focus on studying the effects of adaptive learning using the detection of ErrP as feedback to the classifier.

The block diagram of the system is shown in Fig. 1. The chain in the framework starts from a user using an SSVEP-based BCI, whose SSVEP trial is fed to the SSVEP classifier (arrow 1). Then the SSVEP classifier outputs the predicted label of this trial (arrow 2), and the predicted label is displayed on the screen to further stimulate the user (arrow 3). The ErrP trial which contains the user's EEG response to

K.-J. Chiang is with the Department of Computer Science and Engineering, University of California - San Diego, La Jolla, CA 92092, USA (email: kuchiang@eng.ucsd.edu). D. Emmanouilidou, H. Gamper, D. Johnston, M. Jalobeanu, E. Cutrell, A. Wilson, and I. Tashev are with the Microsoft Research, Redmond, WA 98052, USA (email:ivantash@microsoft.com). W. W. An. is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA (email: wenkanga@andrew.cmu.edu).

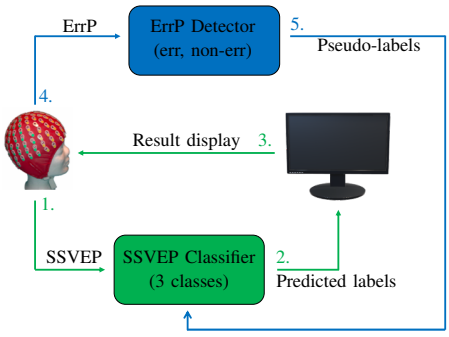

Fig. 1. The block diagram of the proposed closed-loop framework.

the predicted result is then fed into the ErrP detector (arrow 4). Finally, the ErrP detector predicts whether the SSVEP label matched the user's expectation by monitoring whether an ErrP exists in the ErrP trial. If no ErrP is detected, the system assumes that the predicted SSVEP label is correct, and uses the pair of the SSVEP trial and the predicted label (pseudo-label) to further fine-tune the SSVEP classifier (arrow 5, more details in section II-F).

Note that in this study, we focus on the improvement of the SSVEP classifier. Although the ErrPs also have variability across sessions, studies have shown that the detection of the ErrPs can have comparable performance in a cross-session scenario against a within-session scenario [9].

B. Experiments

The experiment consisted of a user interacting with SSVEP targets as a BCI-controlled keyboard. There were three square targets with texts, "Left/Enter/Right" on each at the lower part of the screen of the experiment. At the beginning of each SSVEP trial, a red arrow pointed at one of the targets for 0.5 sec, and then all three targets started to flicker at different frequencies (7.5, 10, and 12 Hz) for 2 secs. The subject was asked to look at the highlighted target during the flickering. A 1-sec pause followed the flashing in which all three targets became black. After the pause, the ErrP trial started. One of the targets turned green and a text "Result: Left/Enter/Right" corresponding to the predicted result of the SSVEP trial was displayed at the center of the screen for another 0.5 sec. Finally, another 1-sec pause was presented as a short break before the beginning of the next SSVEP trial. The arrow pointed at each of the three targets for one time every three SSVEP trials, but the order was randomized. The EEG device used in this study was actiCap Xpress Twist and the LiveAmp with a 500-Hz sampling rate from Brain Products, Germany.

C. Datasets

Following state regulations for mitigating the infection of covid-19, only two subjects participated in the study. There were 8 recording sessions (id 1 to 8) evaluated for a main participant (one of the authors) and 3 (id 1 to 3) recording sessions for a secondary participant (a male from the same household). For the main participant, there were also three pilot ErrP sessions in which the predicted results

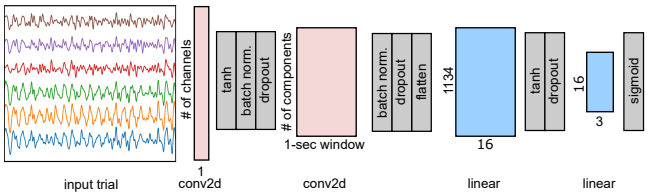

Fig. 2. The structure of the CNN model for the SSVEP classification.

of the SSVEP were randomly generated with the probability $Pr(\text{correct}) = 0.7$ and $Pr(\text{wrong}) = 0.3$ to stimulate the ErrP responses. There was another pilot session, id 0, that included both SSVEP trials and actual ErrP trials. Using sessions id 1 to 8, before the experiment, an SSVEP classifier was trained using the SSVEP trials from the previous session (e.g. trained with session id 0 for session id 1, session id 1 for session id 2, and so on), and the SSVEP classifier was used during the experiment to generate predicted results to further induce ErrP. The ErrP detector was trained with the four pilot sessions and classified the ErrP trials of sessions id 1 to 8 offline.

As for the secondary subject, there was one session, id 1, under the cross-subject scenario. In this session, the SSVEP classifier was trained with the 8th session of the main subject and the ErrP detector was trained with the four pilot sessions of the main subject. There were two more sessions, id 2 and 3, under the cross-session scenario that the SSVEP classifier was trained with his own first session, and the ErrP detector was trained with the three pilot sessions from the main subject plus the ErrP trials from his own first session.

D. The SSVEP Classifier

The raw SSVEP trials are epoched at 0.1 sec after the stimulus onset with a duration of 2 secs. The trials are preprocessed by selecting the channels Pz, P3, P4, Oz, O1, O2, re-referencing to the channel Fz, filtered with a 6-40 Hz band-pass filter, and then down-sampled to a 125-Hz sampling rate.

The SSVEP classifier used in this study is a Convolutional Neural Network (CNN)-based model similar to [10], illustrated in Fig. 2. The model starts with a convolution layer with the kernel size equal to the number of EEG channels $\times$ 1, and the number of the output convolution channels equal to 2. An input trial is first reshaped to dimension 1 (convolution channel) $\times$ 6 (EEG channels) $\times$ 250 (timestamps) and processed with the convolution layer. The purpose of this process is to find the optimal spatial filters. The kernels of the first convolution layer can be viewed as linear coefficients of each channel, and therefore, the meaning of the output is the projection of the trial from EEG-channel-domain to 2 spatial components. Then the output of the first convolution layer is reshaped again to dimension 1 (convolution channel) $\times$ 2 (spatial components) $\times$ 250 (timestamps) and processed with a second convolution layer with the kernel size 2 $\times$ 125 and 9 output channels. The goal of the second convolution layer is to find the temporal signatures within a sliding 1-sec window. Finally, the output of the second convolution layer

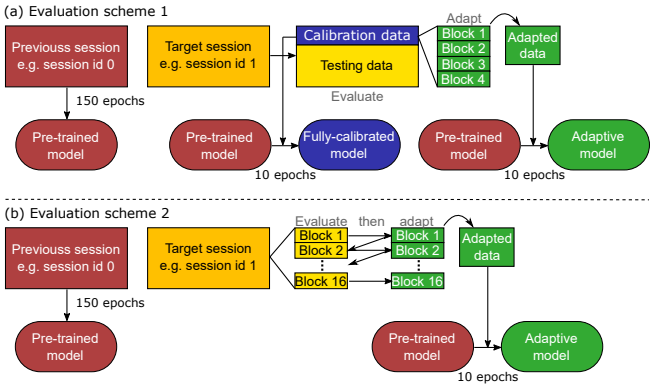

Fig. 3. The flow chart of how the training and testing data are prepared in (a) the first evaluation scheme (b) the second evaluation scheme.

is connected with two linear layers with output dimensions equal to 16 and 3 to project the tensor to class spaces.

When training the SSVEP classifier to classify the SSVEP trials during a target session, the SSVEP trials in the previous sessions were used. Ten percent of trials were randomly preserved as validation trials. The maximum number of epochs was set to 150 but the training could be stopped earlier if the validation accuracy decreased for 4 consecutive epochs. The batch size was set to 12, and the optimizer was stochastic gradient descent (SGD) with the learning rate 0.1.

E. The ErrP detector

The raw ErrP trials are epoched with a 0.4-sec window starting from 0.2 sec after the stimulus. Channels Fz, FC1, FC2, Cz, C3, C4, CP1, CP2, and Pz are extracted and re-referenced to the average of TP9 and TP10. Signals are filtered with a 1-10 Hz band-pass filter and down-sampled to a 250-Hz sampling rate.

The ErrP trials are first processed with XDAWN filters to enhance the signal-to-noise ratio [11]. This process is implemented with the Python package MNE [12]. Each trial is projected into 4 XDAWN components, and each component is further segmented into 2 windows (0-125 and 125-250 ms). Linear Discriminant Analysis (LDA) is applied across each window of all trials, and an LDA index is obtained for each window in each trial. Therefore, for each trial, 8 total feature values—4 (XDAWN components) $\times$ 2 (windows)—are calculated. Finally, these features are used to train a Logistic Regression (LR) classifier (implemented using the Python package Scikit-learn [13]).

F. Adaptive Learning

In this work, we use two evaluation schemes to simulate the effects of the adaptive learning process as the BCI system is being used. These two schemes are illustrated in Fig. 3. The first scheme allows us to compare the performance between our **adaptive model** and the model gone through a full calibration process conventionally. In the first scheme, the data of the previous session is used to train the **pre-trained model** as described in section II-D. The target session is divided into calibration data and testing data with a ratio of $1:3$. The calibration data (with the ground-truth labels) are used to fine-tune the **pre-trained model** to obtain the **fully-calibrated model**. The fine-tuning process consists of training 10 more epochs using the new data with a batch size of 12 and the learning rate and the momentum set to 0.1 and 0.01 respectively for the SGD optimizer. The calibration data are further split into 4 blocks. These blocks of trials are gradually added into the training pool as adapted trials to fine-tune the **pre-trained model** in the chronological order, which simulates that the system is progressively fine-tuned during the online usage. Starting from the first block, for every ErrP trial in the block, the ErrP detector makes a prediction of whether there exists an ErrP in this trial, if the ErrP detector has more than 0.75 confidence that an ErrP doesn't exist, then the corresponding SSVEP trial and predicted label are added into the adapted pool. Every time a new block of trials are partially added into the adapted pool, a new **adaptive model** is trained by fine-tuning the **pre-trained model** with the adapted trials in the pool, and all models are evaluated using the testing data.

In the second scheme, the **pre-trained model** is acquired in the same way, but the target session is no longer divided into calibration data and testing data. The target session is directly split into 16 blocks instead. Similar to the first scheme, each block is also added to the adapted pool in order. However, in the second scheme, models are evaluated with the new block before it is added into the pool. Also, there is no **fully-calibrated model** to compare. This scheme allows us to simulate the online performance the **adaptive model** could achieve.

III. RESULTS

The average results across the eight sessions of the main subject in two evaluation schemes are shown in Fig. 4. The shaded areas indicate the standard error of the accuracy of **pre-trained model** and **adaptive model** across the eight sessions. Several additional curves are added as comparisons in different scenarios which are explained in the legend area. The p-value of the Wilcoxon signed-rank test between the **pre-trained model** and the **adaptive model** is 0.027.

Fig. 5 shows the results of the three sessions of the secondary subject. In the top row, the data used to train the **pre-trained model** are from the last session of the main subject, while in the bottom two rows, the ones are from the first session of the secondary subject as described in II-C.

IV. DISCUSSIONS

As shown in Fig. 4, the **pre-trained model** has the lowest accuracy in most circumstances when other models are fine-tuned with the trials within the target session. The performance of our proposed method, the **adaptive model** increases as new blocks of trials come in, and its accuracy gradually converges to the one of the **fully-calibrated model** with a similar increasing rate compared to the **model with true labels**. This suggests the calibration period for obtaining the **model with true labels** and the **fully-calibrated model** could be replaced with the period of online use. Furthermore,

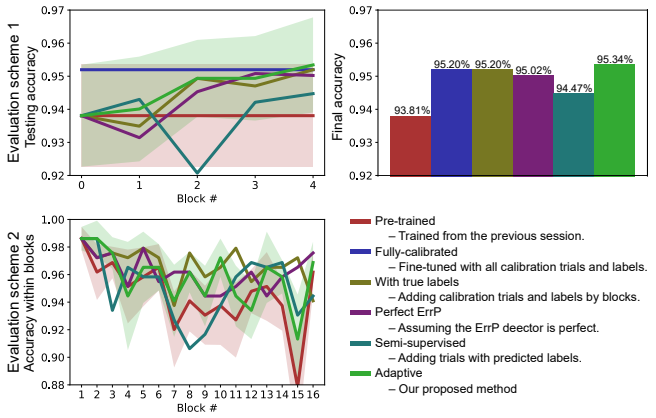

Fig. 4. The results of the main subject under two evaluation schemes. The top left panel shows the accuracy of different models evaluated under the first scheme. The shaded areas indicate the standard error. The top right panel further highlight the accuracy at the point after the fourth block is adapted from the top left panel. The bottom left panel shows the accuracy within each block under the second scheme.

the comparable growth in accuracy between the **adaptive model** and the **model with perfect ErrP** implies that the performance of the cross-session ErrP detection (around 87%) is good enough. In the bottom left panel, we can see that the **pre-trained model** fails to maintain the good performance especially at later blocks potentially due to the fatigue making the signal quality worse. However, the **adaptive model** is capable of keeping decent performance. Also, the similarity in performance between the **adaptive model** and the **model with true labels** validates the success of the ErrP detector. Finally, the **semi-supervised model** which progressively fine-tunes the model with the SSVEP trials and their predicted labels is also compared, and its performance tends to be worse than the **adaptive model**.

In Fig. 5, the results of the secondary subject show lower overall accuracy. The reason for the overall lower accuracy could be subject-variability. However, the trend is similar to the main subject. When looking at the left panels in Fig. 5, the **adaptive model** has comparable performance with the **pre-trained model** after four blocks of trials are adapted. Yet, if the adapting process is prolonged (the right panels in Fig. 5), the performance of the **adaptive model** starts to diverge from the **pre-trained model**. Also, after the fifth block is adapted, we can see a clearer trend that **adaptive model** has increasingly better performance as the **model with true labels** and the **model with perfect ErrP** do, especially in the cross-session scenarios.

The limitations of this study are that the size of the dataset is small, and the performance in the cross-subject scenario can be improved. Still, this work proposes an adaptive framework that could facilitate plug-and-play BCIs.

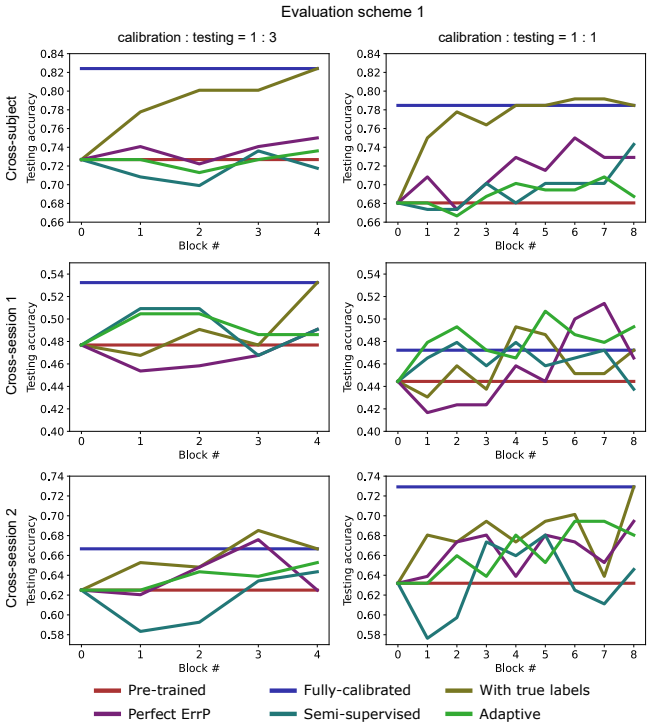

Fig. 5. The results of the secondary subject under the first evaluation scheme. In the left panels, the calibration data and the testing data are split from the target session with a ratio of 1 : 3, while the right panels use a ratio of 1 : 1, which assumes longer calibration/adapting process.

References

[1] M. Spüler, "A high-speed brain-computer interface (BCI) using dry EEG electrodes," *PloS one*, vol. 12, no. 2, p. e0172400, 2017.

[2] M. Nakanishi, *et al.*, "Enhancing detection of SSVEPs for a high-speed brain speller using task-related component analysis," *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 1, pp. 104–112, 2017.

[3] C. Guan, *et al.*, "High performance P300 speller for brain-computer interface," in *IEEE International Workshop on Biomedical Circuits and Systems, 2004.* IEEE, 2004, pp. S3–5.

[4] B. Kerous, *et al.*, "EEG-based BCI and video games: a progress report," *Virtual Reality*, vol. 22, no. 2, pp. 119–135, 2018.

[5] P. Wang, *et al.*, "A review on transfer learning for brain-computer interface classification," in *2015 5th International Conference on Information Science and Technology (ICIST).* IEEE, 2015, pp. 315–322.

[6] A. Buttfield, *et al.*, "Towards a robust BCI: error potentials and online learning," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 164–168, 2006.

[7] A. Llera, *et al.*, "On the use of interaction error potentials for adaptive brain computer interfaces," *Neural Networks*, vol. 24, no. 10, pp. 1120–1127, 2011.

[8] N. R. Waytowich, *et al.*, "Unsupervised adaptive transfer learning for steady-state visual evoked potential brain-computer interfaces," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC).* IEEE, 2016, pp. 004 135–004 140.

[9] P. W. Ferrez *et al.*, "Error-related eeg potentials generated during simulated brain–computer interaction," *IEEE transactions on biomedical engineering*, vol. 55, no. 3, pp. 923–929, 2008.

[10] C.-S. Wei, *et al.*, "Spatial component-wise convolutional network (sc-cnet) for motor-imagery eeg classification," in *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER).* IEEE, 2019, pp. 328–331.

[11] B. Rivet, *et al.*, "Theoretical analysis of xDAWN algorithm: application to an efficient sensor selection in a P300 BCI," in *2011 19th European Signal Processing Conference*. IEEE, 2011, pp. 1382–1386.

[12] A. Gramfort, *et al.*, "MEG and EEG data analysis with MNE-Python," *Frontiers in neuroscience*, vol. 7, p. 267, 2013.

[13] L. Buitinck, *et al.*, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.