BME 296: BCI Project 2: SSVEP

Spencer Dooley, Alex West

11/9/2021

**Introduction**

In this research project, we investigate the use of steady-state visual evoked potentials (SSVEPs) in a brain-computer interface. With this goal in mind, a python script was created to take in any given set of epoch FFTs and an electrode of interest to calculate predicted labels representing the predicted stimulus frequency for each trial. Our code is essentially designed to decode EEG signals into predicted stimulus frequencies which will result in an action from the BCI itself. In order to fine-tune this BCI and make it as accurate as possible, many different time windows of epoching data were tested and two figures of merit (classification accuracy and ITR) were compared. The time window that has a good balance of high accuracy and high ITR will be implemented in the final BCI to make it as accurate and fast as possible.

**Methods**

The code included in this project comes with a script containing all modules that would be of use for a user, as well as a test script implementing the aforementioned modules. Starting with the first module seen, *get\_predicted\_labels()*, the user can give any set of EEG epoch FFTs along with the corresponding frequencies, an electrode of interest, and the low and high stimulation frequencies and a list of predicted labels are returned. The list of predicted labels will contain labels representing the predicted stimulus frequency; thus, the length of this list will depend on the number of trials being performed. The second module, *get\_truth\_labels()* is a simple function that extracts the labels representing what the frequency actually was. The following module, *calculate\_accuracy()* takes in the predicted and truth labels calculated by the previous two modules, and compares them element wise to find the proportion of predicted labels that are the equal to their corresponding truth labels. This number represents our accuracy as a proportion and, if multiplied by 100, represents the accuracy as a percent. We use this method to quantify how accurate our BCI would be at identifying different stimulus frequencies. Similarly, the next module, *calculate\_itr()*, takes in our calculated accuracy and the trial (epoching) duration and uses the below two equations to calculate the ITR in bits per second.





The next module is used for the fine tuning of our BCI parameters and helps us choose an epoch time window that balances accuracy and ITR. In *test\_epoch\_limits()*, the user can input an array of start and end times they wish to test. In the module itself, the accuracy and ITR are calculated using the previous modules for every acceptable start and end time combination. We say acceptable start and end time combinations as we cannot have an end time before a start time. In our code, we handle this scenario by inserting a NAN value in for each unacceptable time window. Ultimately, this module will return an array containing the accuracies for each start and end time combination and an array containing the ITRs for each start and end time combination. In our final module, *plot\_results()*, we use pseudo color plots to visualize the two arrays returned in our last module. These visualizations act as a sort of heat map, and quick look into where our figures of merit have high/low values.

**Results**

Chart, histogram

Description automatically generated*Figures 1.1 and 1.2 (Notes: NAN values are shown in white)*

As we see in *Figure 1.1*, the bottom right corner, where end times are significantly larger than the start times, is where we find the highest accuracy for Subject 1 on the Oz channel. For the most part, end times that are very close to start times, do not result in a high accuracy. This makes sense as, there needs to be enough of a time window for adequate data to be collected. On the contrary, in *Figure 1.2* we see very low ITRs in the bottom right corner, and higher ITR values where the time window is shorter. Once again, if we think about the formula this makes sense because dividing the trial duration to get ITR in units of time, determines the magnitude of the ITR (i.e., short time windows mean we divide by a smaller number).

**Chart, bar chart, histogram

Description automatically generated***Figure 2.1 and 2.2 (Notes: NAN values are shown in white)*

Here in *Figure 2.1* and *Figure 2.*2 we are showing the same figures of merit as before but now for Subject 2. We see the same overall pattern with the metrics and certain time windows; however, the metrics themselves are very different here. In *Figure 2.1* we see high accuracies (100% accurate) for more time windows than we saw for Subject 1. For Subject 2, several time windows can be identified that have high accuracy and a high ITR. Overall, it appears the data was better and easier for our program to decode for Subject 2 compared to Subject 1.

**Discussion**

The research question we set out to answer was, *what epoch start, and end times should we use for our BCI for each subject?* From our results, we can begin to make an inference into answering this. A key assumption we are making is that it is important to balance the correctness (accuracy) and speed (ITR) of our BCI. After performing tests of all possible time windows, it is clear that a long-time window results in very high accuracy but a very low ITR in bits/second, while a very short time window mainly results in low accuracy but a high ITR. With the evidence we have attained from this project, we recommend that for Subject 1, a start time between 7.5 and 10 along with an end time between 15 and 17.5 seconds should be chosen and for Subject 2, a start time of 10 seconds and an end time of 12 seconds should be chosen. The ranges for start and end times were chosen for Subject 1 because the accuracy ranges from about 85-90 percent accurate and ITR values range from about 0.1 to 0.15, which are not as low as the majority of ITR values for Subject 1. For Subject 2, the specific values for start and end times were chosen because with that time window, we attain ~90% accuracy as well as an ITR close to 0.5 (one of the highest ITR values we see for either subject). If we had more than two stimulation frequencies or included a rest option, our process to epoch the data would likely change as we would have either a different number of trials or different distribution of trial types. We would also then be predicting labels for three different frequencies rather than two which could cause the accuracy to decrease overall. Thirdly, our ITR calculation incorporates the number of classes which would change and decrease the ITR overall. To avoid accuracy decreasing significantly we would likely have to lengthen the epoch time window. While this almost certainly implies our ITR would decrease given the increase in classes and larger time window, however, if we favor a high ITR, our accuracy will suffer dramatically. It is also worth noting, in our experiment here, all the ITRs are on the same order of magnitude, so a change in ITR will not necessarily be noticeable whereas a decrease from 90% accuracy to 50% accuracy is notable. In conclusion, we favor a BCI built that has start and end epochs around the middle of the event duration for both subject, with a wide enough window to capture enough data, but small enough window to improve ITR.