**OPTIMIZING EEG-BASED CONTROL**

**TO**: Aaron Maus

**FROM**: Justin Haysbert, Gabriel Sagrera, Shayne Shelton, Ryan Stevens

**DATE**: 11/09/23

**GitHub**: <https://github.com/stevensryanw/BCI_Infinity>

**CODE UPDATES**

*Machine Learning Setup and Use*

We have been learning and working on multiple different types of packages for ML, including Sci-Kit Learn, PyTorch, and TensorFlow. To test each model type and its accuracy, we needed a dataset with a similar format to the data that the OpenBCI headset provides. Luckily, when downloading their GUI, we found two sample datasets, one for concentration/meditation and the other for which we will use blinks and jaw clenches.

Next, we needed to import and pre-process the chosen data, and we have created two different ways of doing this. We decided to format our data in a CSV with no header; columns 1-8 are the EEG channels, and columns 9 to 11 are the alphas. The first method simply creates an array of our three alphas; we then sum the three values together for each row and put it in a new array. With this new array, we now loop through, and if the change in alpha is greater or less than 0.001, we will add a spike\_up or spike\_down label to the original formatted dataset; otherwise, we add norm. For our other method, we formatted the data the same way as before, and the difference is in how we create our labels. In this method, we start by creating an array the size of our data that is one column of all norm labels. Next, we use a loop and random to add 40 blink labels throughout the labels data; these labels also randomly last 30-60 samples long. To mimic a rest period at the beginning of the data, we don’t loop through the first 1000 samples.

With these two methods of preprocessing data and adding labels, we could move on to modeling and testing the best Python package. We first worked on SciKit-Learn, a basic machine-learning package with many useful pre-existing functions. In trying SKLearn, we used SVC (Support Vector Classifier), LDA (Linear Discriminant Analysis), RFC (Random Forrest Classifier), GBC (Gradient Boosting Classifier), ABC (AdaBoost Classifier), and lastly, a Voting Classifier for all of the previously stated models.

PyTorch and TensorFlow are the other Python packages we have been using and testing. These packages allow us to use GPUs and create more complex models, including neural networks. Our work with these packages is still in the beginning phases, but we have gotten both to work with very basic sequential models using the M1 processors GPU and with AMD Radeon GPUs.

*Graphical User Interface*

In deciding how to create our all-in-one application and what framework to use, we went back and forth between using a GUI or WebApp/WebUI. At first, we started looking into Django and React, but looking further into it, we found that a WebApp would be overcomplicating the application. Once we realized this, we moved toward a Python-based GUI framework. We are currently working on creating a Tkinter Python GUI to do this. We are all still learning TKinter, so the current implementation is very basic, with only three pages and no existing functionality. The plan is to have a page for live feed viewing, user training, modeling, and outputting predictions to the application (games, wheelchair, robotic arm, etc.). To do this, we must create multiple other Python scripts for functions like plotting the data for the live feed, outputting controls, reading and recording data, prompting user training, and the model selection and modeling itself.

*Python Video Game*

In our all-in-one application, we plan to include a very basic game with directional controls that will allow us to test models and training interfaces before other applications like the wheelchair or robotic arm are ready. Currently, we have something in Unity that is a prototype of the needed functionality. Very little information exists on including a Unity game in a Python GUI. To deal with this, Professor Hassan has recommended that we look at Gather Town and see if there is anything we can use and implement from there. If not, we plan to make the basic game from scratch that is Python-compatible, and when that is done, we may add creative functionality to the game.

**EEG PLANNING & PROCEDURE**

While we still need to receive our Open BCI EEG headset, we have completed a procedural plan for our data collection sessions. To begin this procedural plan, we state the narrative to give a scope to our project, “1 in 50 Americans live with paralysis. Computers can interpret electroencephalogram (EEG) data in real-time to allow users to control devices in real time. EEG data interpretation provides a new way for patients to practice autonomy and increases the accessibility of daily activities. However, current methods using EEG data require multiple, long training sessions, and once trained, they only work for a specific user. In this project, we seek to optimize the training process using new machine learning algorithms and techniques to decrease overall training time and increase the number of people a trained program will work on.”

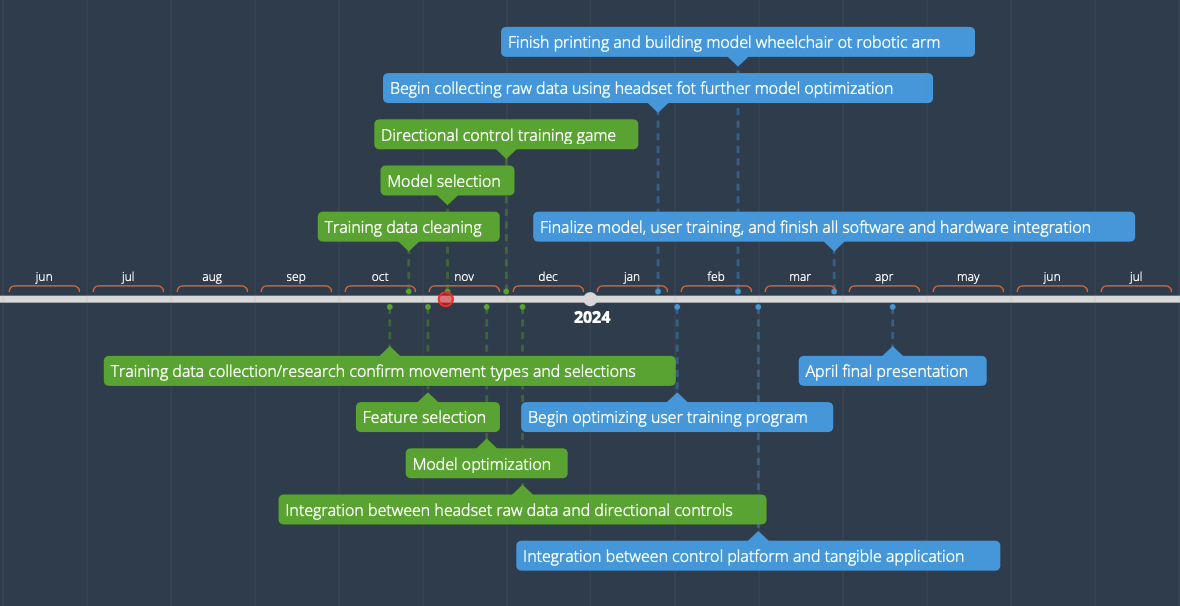
Next, we have outlined a note for our subjects to read before any data collection sessions stating that “On collection days, subjects must be well rested, having received at least 7-8 hours of sleep, must have refrained from caffeine consumption in the last 12 hours, must have refrained from alcohol consumption for the last 12 hours, and must verbalize whether or not they have experienced a state of heightened emotion in the 24 hours before testing.” Upon arriving at the data collection session, the subject must complete a pre-testing survey of the following questions, “Have you consumed caffeine in the last 12 hours? Have you consumed alcohol in the last 12 hours? Have you received at least 7-8 hours of sleep? How would you describe your emotional state over the last 24-hour period? Do you find yourself capable of participating in training today? do you consent to training today?.” Upon completion of this survey, we will review the results, and if they adhere to the subject notes stated above, the data collection session will begin.

The procedure of the collection process reads as follows, “Offline training will occur on Tuesdays during meeting time in an available B School classroom (preferably the same). In preparation for testing, the subject will practice mindfulness meditation for 5 minutes. The openBCI EEG headset will be adjusted for the subject's head, ensuring that all eight electrodes make a reliable connection. For baseline testing, the participant will be seated in a chair with their hands resting in their lap, in a relaxed position. After a baseline is recorded, we will give the subject a 1 minute rest period, followed by a mental preparation period. Following this period, the participant will be prompted to perform a task (MI R/L arm, jaw clench, MI leg movement, eye blink). Forty data points per movement will be used to train a classifier. Online training sessions will include the same procedure as offline, but these sessions will involve real-time use of the classifier. We will use the classifier to provide a real-time feed as a directional game. Our GUI will prompt the subject to complete a directional task using one of the trained controls. After completing the task, the user will state any misclassification". We believe we can provide comfortable and efficient training sessions for our subjects using these guidelines and procedures. We also want to preface that these guidelines and procedures are subject to change and adaptation to ensure the best training experience and results. This procedure has also been registered in our IRB, ensuring we are sanctioned to use participants outside our research group.

**HEADSET PRINTING**

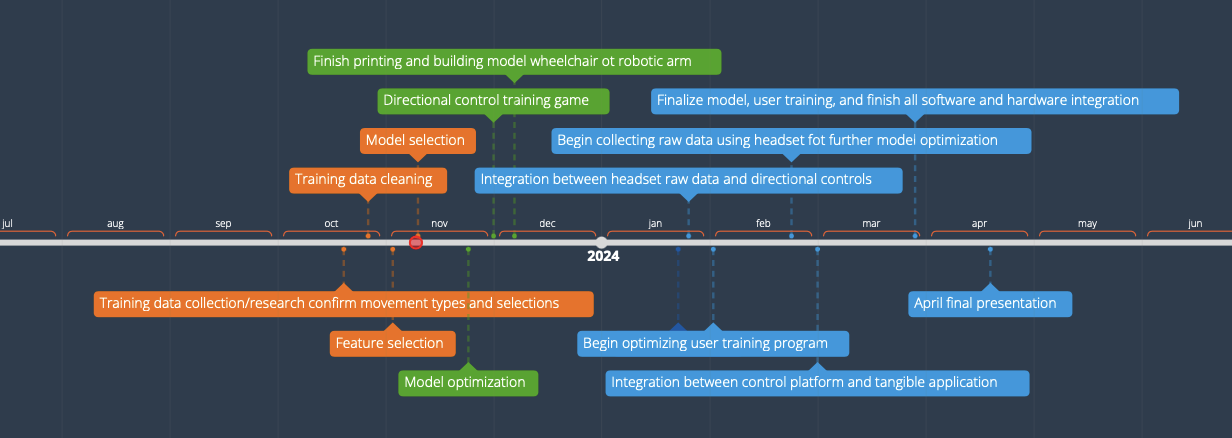
We have decided to use the Ultracortex Mark 4 found on OpenBCI for the EEG. For this headset, the control board and the electrodes are sent, while the actual device to place on the person’s head is 3D printed by the buyer. We have made significant progress on 3D printing the headset for the EEG. As suggested by OpenBCI, we have printed out the headset in four main parts, the front half of the headset, the back half, the board mount and cover, and the screw inserts for the electrodes. For our purposes, we have printed the large headset. There have been some printing issues, specifically for the front and back portions of the headset, due to the size of the print. Due to the headset’s size, there is a significant chance of an error occurring in the print, such as a support not being printed properly, the filament head becoming misaligned, and the spaghettification of a piece of the headset. Due to these issues, the headset halves have been printed several times until a workable piece has been printed. After we printed the headset pieces, we assembled the full headset by gluing the two halves of the headset together, gluing the screw inserts into the areas for the electrodes, and screwing on the board mount. We have only created the large headset version as the person we will first collect data on using the EEG falls under this size, but we will 3D print the other headset sizes if necessary.

**TIMELINE UPDATE & NEXT STEPS**

****

***Figure 1*** - Original project timeline

Our mentor, Professor Hassan, agreed to purchase an OpenBCI headset for our group to have more data collection channels. The headset will take a few months to order and deliver, so we made a few changes to our timeline. The new timeline pushes aspects of our data acquisition to the Spring as we wait for the headset. Additionally, we will begin building our model wheelchair earlier than expected. To leave room for research and optimization in the Fall, we will develop our headset frame and model wheelchair this semester.



***Figure 2*** - Updated project timeline

In *Figure 2*, all completed steps of our timeline are indicated in orange. By the end of this semester, we hope to continue refining the machine learning model, develop a directional control game that integrates into the GUI of our choice, and have a working wheelchair that uses four directional controls.

**TEAM MEMBER RESPONSIBILITIES**

*Justin Haysbert*

Completed a procedural design for data collection and submitted a pending IRB. I am working with openBCI to order the EEG headset. Because headset components have yet to arrive, I will begin working with Ryan to model GUI and finetune classification models. When the headset comes, I will collect data and train new classifiers with Ryan.

*Gabriel Sagrera*

I am currently assisting Shayne with designing the wheelchair model as well as the mechanism for moving it. I will also work on integrating the wheelchair with the headset once both are made.

*Shayne Shelton*

Developed a low-fidelity prototype of the directional control game but will continue researching other game development programs and open-source games to integrate into our GUI. Additionally, I researched small Raspberry Pi-based vehicle models for the model wheelchair concert validation testing in the Spring. We plan to build a working prototype by the end of the Fall semester.

*Ryan Stevens*

Currently working on code to do with the modeling, GUI, and directional control game. The GUI is a very basic three labeled pages with working buttons but no actual BCI functionality. We will be adding each need function over the rest of the semester. The models are currently at a good stopping point where we can focus on the GUI and eventually add modeling functionality. Besides the modeling, we must add user training prompting, plotting live data, outputting predicted controls, etc.

**MENTOR MEETINGS:**

We have met weekly with Professor Hassan every Friday at 2 pm. In these meetings, we have discussed the week’s progress, the timeline overall, and what to do for the upcoming week. We hope to have the headset soon for testing, in which we can work on the code during the meetings instead of only going over deadlines.