A black background with a black square

Description automatically generated with medium confidence

Training EEG Brain Wave Data with Machine Learning to Output Directional Controls

<https://github.com/stevensryanw/CogniSync>

Justin Haysbert, Gabe Sagrera, Shayne Shelton, Ryan Stevens

Department of Computer Science, Tulane University – New Orleans, LA

CMPS 4020 Spring 2024

Faculty Mentor: Saad Hassan

SUMMARY

*Motivation*

Millions of people worldwide have paralysis. Electroencephalogram (EEG) headsets measure electrical activity in the brain like motor imagery related brain waves. Machine learning (ML) approaches could potentially categorize a user’s brain state into directional controls for wheelchair navigation. However, first-time user training is long and user specific, and EEG’s headsets experience a low signal-to-noise ratio.

*Solution*

This project aims to develop a comprehensive user interface that will streamline recording, modelling, and prediction processes. Researchers may refine and optimize accuracy while reducing training time in future studies. Simultaneously, the capabilities of EEG headsets will be more accessible to broader audiences.

*Prior Work*

Prior work on EEG-based brain-computer interfaces (BCI) control is sparse, but many applications involve solutions for those with mobility disabilities. For those with disabilities, tools like wheelchairs, mechanical arms, and prosthetic limbs are very important. BCI could provide interaction control over these devices to those who cannot utilize joysticks or controllers. Essentially BCI helps solve human-computer interaction (HCI) problems. EEG is one of the most used types of BCI because it does not require invasive techniques and can provide high temporal resolution.

Huang et al. (2019) developed an integrated system utilizing EEG and EOG signals to control both a wheelchair and a mechanical arm. By decoding imagined hand motions and eye movements, users could steer the wheelchair and manipulate the mechanical arm with notable accuracy. This study underscored the feasibility of non-invasive EEG-based control systems in facilitating seamless interaction with assistive devices, offering a promising avenue for enhancing mobility and autonomy.

Jeong et al. (2020) adopted a deep learning approach to improve BCI performance for controlling a robotic arm. Their innovative use of a multi-directional CNN-BiLSTM network allowed for better recognition of user intentions in three-dimensional space. Emphasizing the need to minimize cognitive workload on users, the study suggested leveraging pre-trained models to streamline calibration processes. By optimizing training environments and harnessing deep learning techniques, this study highlighted avenues for enhancing the reliability and efficiency of EEG-based BCIs in real-world applications.

Furthermore, Song et al. (2020) explored a novel method based on the P300 EEG component, achieving impressive accuracy rates in controlling an assistive robot. By leveraging raw EEG data and innovative processing techniques, they demonstrated the potential for real-time control methods with high accuracy. This approach holds significant promise for individuals with severe mobility impairments, offering a means to interact with mechanical devices effectively and independently.

Additionally, Bousseta et al. 2018 introduced a sophisticated EEG-based BCI system capable of controlling a robot arm through imagined motor tasks. By recording EEG signals from the scalp and employing advanced signal processing techniques, users could translate their intentions into precise commands for the robot arm. The study highlighted the importance of user adaptation and learning in optimizing BCI performance, showcasing the potential for EEG-based BCIs to empower individuals with disabilities to regain control over their environment.

These studies illustrate the versatility and efficacy of EEG-based BCIs in facilitating human-computer interaction, particularly in the realm of assistive technology for individuals with mobility impairments. By leveraging innovative signal processing methods, machine learning algorithms, and user-centered design principles, researchers continue to advance the capabilities of EEG-based BCIs, offering new possibilities for enhancing the quality of life for individuals with disabilities.

*Methods & Tools*

After months of in-depth literature review on EEG controllers, the team began building their product.

The team hoped to utilize Sci-Kit learn, Tensorflow, and PyTorch modules for their machine learning modeling, so they opted to develop their product in Python. They used Custom TKinter for UI development since it is a Python based package that allowed for easy integration of other scripts which can handle prompting and modelling. Both Sci-Kit learn and PyTorch are fully implemented into our GUI and can be used on recorded data with labeling. To create the data used the GUI has a user prompting page that with prompt user selected movements and annotate the data with those movements as labels. Depending on the user’s device PyTorch will check for available devices and use either the M series ‘MPS’, Nvidia GPUs ‘CUDA’, or just any ‘CPU’.

Throughout the GUI to record, predict, or output controls we implemented Python threading. In user prompting we have a recording thread to record and create our annotated data. The team tested their models by outputting controls to a basic computer game via the Snake Game page and to a RC car via the USB output page. The snake game was completely developed using functions and classes from CustomTKinter, and the team integrated key bind controls to make it more seamless. In the Snake Game or USB output pages we again have that recording thread to read the headset data. In addition, we have a prediction thread that takes the data and uses the selected model to predict on that data. Lastly, we have a streaming thread that takes the predictions made and outputs them as a directional control for either the Snake or USB output.

The model wheelchair parts (batteries, motors, frame, microcontroller, etc.) were funded by Professor Hassan. The team opted to use a WiFi-based microcontroller called an ESP8266. Using online scripts as inspiration and Arduino IDE, the team programmed the ESP8266 to control the motors based on keyword inputs it receives from the network. This allowed for the UI to communicate across the network directly to the controller via integrated Python scripts and the controller’s unique IP address.

*Results*

The team developed a comprehensive user interface that allowed users to pick movements, train using a built-in prompter, and build a machine learning model using an algorithm of their choice. The program allows flexibility in allowing users to pick their own movement types and the number of movements they would like to train their model with.

A screen shot of a graph

Description automatically generated

Left Arm ----------

Right Arm ----------

Legs ----------

Jaw Clinch ----------

\*Vertical lines denote new movement type\*

*Figure 1* - Sample 8 channels EEG recording where vertical dashed lines show movements and the horizontal lines are the EEG channels

The team ran into a slight set back toward to end with livestream their predictions directly to their mechanical device; however, the recording and model aspects of their design we fully functional enough to success collect 3 datasets from one of the members and train that dataset using 6 different models.

A graph showing a number of data

Description automatically generated with medium confidence

*Figure 2* - Trial-by-trial Optuna hyperparameter optimization for the PyTorch machine learning library measuring accuracy, f1 score, precision, and recall

A graph with numbers and a bar

Description automatically generated with medium confidence

QDA = quadratic discriminate analysis

LDA = linear discriminate analysis

RFC = random forest classifier

DTC = decision tree classifier

*Figure 3* - Optuna optimal hyperparameters for each machine learning approach used on the collected EEG data, measuring accuracy, precision, and recall. Values displayed reflect the best model generated using that machine learning approach.

After optimizing hyperparameters, the team noticed that random forest classifier and decision tree classifier performed the best on the data they collected. For future work, we recommend users to use these methods. However, the team is aware of the possible impacts of overfitting. Future work on more datasets is required to make a more definite conclusion.

*Significance*

The GUI developed by the team can help others who are interested in this area of research. It may also be more practical as the current state of EEG use for BCI is not consolidated and is inconsistent in data collection and processing practices across previously cited sources. By creating a customizable GUI that has a standard data collection and processing, it allows for greater collaboration between groups and research in the area to be conducted faster as it does not require new processes to be developed each time. This streamlined process also allows more people to have access to this technology allowing for more personal use of EEG BCI.

Our models for random forest classifier, decision tree classifier, and PyTorch neural net had promising scores in the metrics we used but did not translate as well as hoped into live model prediction and user control. When the live model prediction was implemented, we discovered that those models, as we had fitted them, did not always accurately predict movements during this. There are multiple possibilities as to why this could be, e.g., overfitting, not calibrating the predictions in a new environment, and the headset not being positioned in the same way on the user’s scalp. This does limit the GUI as the preset parameters we have for the models may not create an accurate model for the user and will likely require others to optimize the models further limiting the accessibility of the GUI. The GUI itself is also limited as we only had access to one version of the OpenBCI headset limiting what headsets could be used with our GUI currently.

PROJECT REFLECTION

*Future Work*

While we are satisfied with the progress made on this project, we foresee many possible extensions, modifications, and future works. For starters, we were not able to have our models give accurate predictions for all four movements. There are several reasons why this could have occurred, the first being data preprocessing and feature extraction. These are in addition to the reasons such as overfitting and hardware calibrations issues.

*Data Preprocessing*

While we utilized pyOpenBCI’s data streaming capabilities to preprocess our datasets we do believe that more preprocessing could be done. It is very possible that our data contained unwanted noise and useless information. While proper data cleaning can be tedious when it comes to brain signals, additional filtering of noise could prove useful. Feature extraction is another method that could be implemented into our preprocessing pipeline to reduce the dimensionality of the data so that it accurately represents the underlying signals ([Krishnan and Athavale, 2018](https://www.frontiersin.org/articles/10.3389/frai.2022.1072801/full#B19); [Krishnan, 2021](https://www.frontiersin.org/articles/10.3389/frai.2022.1072801/full#B18)).

*Larger Datasets/Dataset Appending*

Another reason for our models' low predictive accuracy could be due to lack of data. Our models were trained on 4 movements repeated 40 times. It's very likely this was simply not enough data to build a useful model on. Building larger datasets is essential to multiclassification projects and we have provided a platform just for that. Our GUI allows users to easily record multiple EEG sessions. While our platform makes it possible for users to adjust the amount of data they are collecting, these bigger sessions would be long and daunting. It is more likely that users will want to collect chunks of data at a time. To accommodate this need, a feature allowing the augmenting multiple datasets could prove useful. This could allow for the consolidation of a user's recording sessions overtime.

*Online Training*

Adding an online training option could also alleviate the need for more engaging data collection methods and the snake game could easily serve this purpose. Since normal training sessions are long and boring, the snake game could substitute this with short interactive training sessions. The snake game could be used to validate the model on the fly while appending new data to the dataset. However, this feature is not limited to only the snake game; any game that uses basic directional controls would work. This would give users a variety of ways to train online, leading to a much higher incentive toward data collection.

*Model Fine Tuning*

In future work users can use Optuna to fine tune individual model parameters for Sci-Kit Learn, Tensorflow or PyTorch. Then users can take the results of those trials and manually input them into model\_bci.py. Our previous Optuna scripts for Sci-Kit Learn and PyTorch are in our GitHub and can be used to tune whatever model is chosen.

*Aesthetics & Quality of Life*

While our GUI is in a working state there are some improvements that can be made for aesthetics and quality-of-life. In terms of aesthetics the GUI could be made to look a bit more modern and appealing. As users in the modern era of technology we like to look at pretty products. Webpages that look good and are easy to use will gain more traffic and draw more user retention. The process of collecting EEG data is already boring, but boring and ugly is not acceptable. Additionally, some design upgrades could be made such as a user login system, a headset calibration feature, more data visualization features, user personalized file storage, and additional user recording customization features. Our GUI currently contains one bug which is the resetting of Tkinter frames. We found it difficult to completely wipe pages that have been used to collect or edit data. Currently, the GUI needs to be restarted for every data collection session. This is inconvenient and quite frankly a pain. A resolution to this bug would drastically improve quality of life.

*Impact and User Accommodation*

Our project has provided an open-source interface for anyone interested in building brain-computer interface models to begin their work. Now, the two main points of consumer access to brain computer interfacing are OpenBCI and Emotiv. These companies both supply headsets but OpenBCIs software is not comprehensive enough to provide modeling features and Emotiv locks their data collection and modeling software behind expensive month-to-month paywalls. Our project's open-source nature allows anyone using an OpenBCI headset to easily collect, preprocess, visualize, model on, and predict EEG data. This allows both beginners and experts to quickly build large datasets with multiple customization options and grants the ability to utilize pre-built machine learning models on said datasets. Beyond that we also give access to validation methods such as the snake game and built-in connection to output devices such as our model wheelchair. This application serves as a great launching point for future projects in the realm of EEG based brain-computer interfacing and serves to streamline the process of building these complex systems.

PROCESS

*Technical Reflection*

Before beginning the project, the team did a deep dive into prior work on EEG data. While they waited on their headset, they tried to use data collected from previous experiments. In doing so, they learned what the data looked like and how to manipulate it. It also gave them a baseline for developing our training protocol. The project would not have been possible without machine learning model Python packages as these streamlined the modeling process. Sci-kit learn, Neuropype, and PyTorch were critical for modeling and Optuna helped with optimization. All this prior art laid a foundation to make the project possible.

During our initial research, we sought to build a dataset using open-source EEG data present on the internet. We quickly found that these datasets were not organized in a structured manner and that each dataset varied in EEG electrode placement. This placement is imperative to the data collection, as the movements we are recording directly relate to the brain regions we are reading from. Due to this we decided to begin building our own datasets but needed an interface to do so. So, we made the decision to build a complete UI for data collection and modeling.

In designing our GUI, we considered the current difficulties in creating an EEG dataset. The first was just adding the labeling and annotations itself with our prompting page. Next, we thought that a user may want to visualize the data, so we created a new page to simply view the data with multiple levels of customization. Once we solved the data collection and visualization challenges, we got to work on a modeling page in which we allow customization of the data and model used. Lastly, we had to make the pages for output or proof of concept. To start we chose a simple snake game with no lengthening snake and no boundaries. In addition, we also created a page that outputs the controls it receives to our scale robotic wheelchair. In creating this GUI, we believe we have created the first open-sourced application with prompting, modeling, and predicting using EEG.

In completing this Capstone project, we have all learned crucial lessons to working on and running a project. Time management was a valuable skill throughout the year, because we did have the class milestones, but still needed to work as a group on our own to stay on track. Additionally, leadership showed that is was essential to successfully complete the GUI. To properly manage the project we needed to meet multiple times a week in our lab and set milestones for our weekly meetings with Professor Hassan. In addition to time management and leadership, we found that the team needed to properly communicate. We found that at any point if we lacked communication or meetings, the project would fall behind and deadlines would become tight. Additionally, we found that for a new and novel idea ample research is crucial to a head start or a fast start.

When looking ahead into project management and product development, the necessary process includes ample research ahead of time. Once the research has been done a team needs to set up the right controls and positions. Additionally, time management and communication are crucial to keeping the team together and on track. Lastly, a team needs to have a leader that lacks bias and has the greater good of the team in mind. A selected leader should be passionate about the team's mission and always contribute to the project. Lastly, we think it is crucial to any projects motivation to stay curious and keep everyone motivated and on the same page.

*Professional Reflections*

Some things we learned for group management was to have a regular meeting time set so that we can keep each other on track, divide tasks, and catch each other up on progress so that everyone on the team knew what needed to be done personally and overall, as a group at any time. During these meetings we would often discuss issues that were brought up by someone and collectively work through errors or brainstorm ideas for the issue to solve it. We would also each update the group on our progress which encouraged accountability, but the group was consistently open to discussing revision of the deadline if necessary, allowing for open and honest communication in the group.

We also learned the essential skill of presenting ideas to our peers. Peer-to-peer communication is key in computer science as it fosters collaboration and the exchange of ideas. Our in-class presentations have allowed us to sharpen these skills. We were also given the opportunity to present in a professional setting during the AI Future Minds Expo. During these presentations, we were required to present our ideas clearly and precisely. We learned that no matter how knowledgeable you are about a topic if you can't communicate this knowledge, it's useless. Our capstone expo presentation gave us the opportunity to present the complex ideas behind our project to a general population without a computer science background. Through this we learned that it's important to gauge your audience's knowledge on the subject and to speak to them at their level of understanding.

Our objective originally was to provide a new method of wheelchair control for paralysis patients to open more avenues for greater independence in mobility and device control. By streamlining and consolidating the process, we have made it more accessible to all audiences allowing more people to use and develop the capabilities of EEG use, BCI development, and machine learning. We have given another avenue to accelerate research for EEG and BCI use allowing for more research to be done and in the longer term, help develop and refine the accuracy and practicality of it. Creating the GUI so that a variety of data and movements can be used allows for more people since it broadens the use case of our GUI further increasing the accessibility of the technology.

REFERENCES

*Bousseta, R., El Ouakouak, I., Gharbi, M., & Regragui, F. (2018)1. EEG based brain computer interface for controlling a robot arm movement through thought2. IRBM, 39(2), 129-135.* [*https://doi.org/10.1016/j.irbm.2018.02.001*](https://doi.org/10.1016/j.irbm.2018.02.001)

*Huang Q, Zhang Z, Yu T, He S and Li Y (2019) An EEG-/EOG-Based Hybrid Brain-Computer Interface: Application on Controlling an Integrated Wheelchair Robotic Arm System. Front. Neurosci.13:1243. doi: 10.3389/fnins.2019.01243*

*J. -H. Jeong, K. -H. Shim, D. -J. Kim and S. -W. Lee, "Brain-Controlled Robotic Arm System Based on Multi-Directional CNN-BiLSTM Network Using EEG Signals," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 28, no. 5, pp. 1226-1238, May 2020, doi: 10.1109/TNSRE.2020.2981659.*

*Song Y, Cai S, Yang L, Li G, Wu W, Xie L. A Practical EEG-Based Human-Machine Interface to Online Control an Upper-Limb Assist Robot. Front Neurorobot. 2020 Jul 10;14:32. doi: 10.3389/fnbot.2020.00032. PMID: 32754025; PMCID: PMC7366778.*