Development of an automated high-throughput machine learning pipeline for phenotypic classification based on rodent EEG frequency spectra

Fangfang Sheng[1], Daniel J. Graziano[1], Eric J. Ma[1], Mei Xiao[1], Holger Hoefling[1] and Michelle M. Sidor[2]

[1]Informatics Systems and Neuroscience Department, Novartis Institutes for Biomedical Research, Cambridge MA 02139

[2]DeepCure Inc, Boston MA 02203

Introduction

- Electroencephalography (EEG) is a non-invasive and translational method to record brain activity across species, with robust alterations in the EEG frequency spectra found across several neurological diseases.
- Conventional quantitate EEG analysis divides spectra into pre-defined bins (alpha, theta, etc.). Subtle changes that fall between bins could be missed. This problem coincides with the low signal to noise and limited size and availability of most existing EEG datasets.
- Leveraging machine learning tools to select features in an unbiased manner, even on smaller pre-clinical datasets, would be immensely helpful in building more robust translatable biomarkers.
- Despite the advances in artificial intelligence, signal processing and interpretation in most cases is still performed manually. The robust automatic classification of these signals is an important step towards making the use of EEG more practical in many applications and less reliant on trained professionals.
- Here, we present a modular, automated high-throughput ML pipeline for rodent EEG spectral analysis which is divided into three parts: feature extraction, feature selection, and classification.

Keywords

EEG ; quantitative EEG (qEEG) biomarkers; Classification algorithms; Feature Selection Algorithms; Automated phenotyping

Generation of Features from raw EEG signal

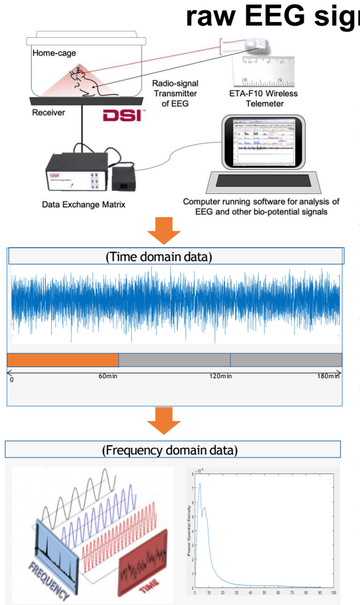

Figure 1. Data Sciences International (DSI) wireless radiofrequency telemetry platform was used to obtain EEG recordings in rodents. EEG time-series data is comprised of multiple frequency components. After pre-processing (rejection of animals with bad signal) and filtering (0.5 Hz hi-pass, 150 Hz low-pass), a custom script deconvolved the data from the time-domain into the frequency domain using the Fast Fourier Transform (FFT). The resultant spectra were used directly to create the features as well as the canonical frequency bins obtained from said spectra.

Machine Learning Pipeline

Figure 2. The **machine learning workflow** has been divided into three essential stages: feature extraction, feature selection and classification.

Feature Extraction

Time segmentation The PSD spectra estimate is calculated by overlapping windows according to Welch's method. The output for this can be averaged in the time domain with any level of granularity. As an example case for this project, the first 60 minutes of the phenotyping recording session were used, but any time bin selection (or combinations of multiple time bins) can be used in this pipeline.

Binned and unbinned full spectrum power spectral density are calculated and normalized to the overall spectral power. There are two ways to generate features from the relative EEG power spectral density, traditional frequency bin definitions ("Binned Spectrum = Delta, Theta, Alpha1 & 2, Sigma, Beta1 & 2, Low, Med, High Gamma, Gamma) and an unbinned approach (Full Spectrum = 2Hz resolution). Each of the frequency bins is treated as a single parameter and tested for its ability to separate groups; 53 such parameters (frequency bins) were available to be selected.

Feature Selection

ANOVA F-value filter method was used to determine if the means of two or more groups that are significantly different from each other using various combinations of features To calculate the F statistic, the ratio of variance between the groups and variance within the groups was taken.

$$Fvalue = \frac{\frac{SSB}{df_B}}{\frac{SSW}{df_W}}$$

Where SSB was the Between the Sum of Squares, SSW was the Within Sum of Squares, df was the degrees of freedom.

Jax-based logistic regression with lasso attention uses the LASSO method on logistic regression model and exclude the features with coefficient equal to zero as is a powerful method for feature selection. We applied a two-layer Neural Network structure.

XGBoost is short for Extreme Gradient Boosting and is an efficient implementation of the stochastic gradient boosting machine learning algorithm. We imported XGBClassifier from xgboost package for python.

Recursive Feature Elimination(RFE) recursively removes features, builds a base comparison model using the remaining attributes and calculates model accuracy. Features are ranked by the model's feature importance attributes but requires defining the optimal number of features to search for beforehand.

Classification model

Figure 3. Decision Tree, Random Forest, Logistic Regression and Linear Discrimination Analysis are are all ML algorithms suited to the task of binary classification.

Decision trees (DT) use rules nested in a hierarchical tree structure to classify new data. Data is applied one by one to the tree to undertake a classification process by passing it through all of the rules in the tree.

Random forest algorithm (RF) combines multiple trees with multiple combinations of variables, each of which can be trained on different training datasets. This results in many trees being generated, and the ensemble of trees are used in a majority voting fashion to classify instead of generating a single decision tree. Different training clusters are generated from the original training set by using bootstrap and random feature selection.

Logistic Regression(LR) is a classification algorithm used to assign observations to a discrete set of classes. Logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to different groups that are being classified.

Linear Discrimination Analysis (LDA) estimates a linear function of the variables that acts as a classification rule to predict group membership.

Testing and Evaluation

To derive a more accurate estimate of model prediction performance , we used **5-fold cross-validation** combining the results of **1000 runs of the pipeline.** To speed up the process, we analyzed datasets and trained models on an HPC system with Dask, a parallel computing library that integrates well with the existing Python software ecosystem. Also, a streamlit web-application was built to have a user-friendly front-end to testing and comparing the performance of these methods

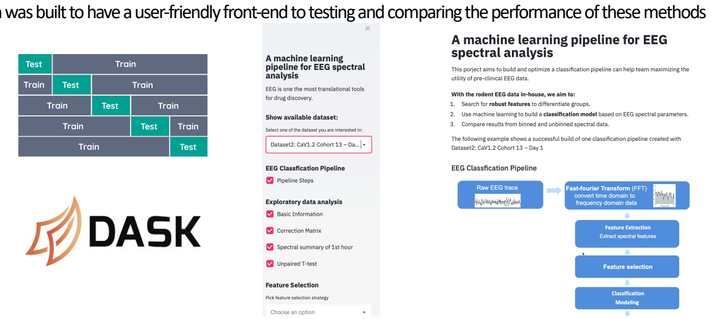

Figure 4. The goal of cross-validation is to test the model's generalization and prevent overfitting or selection bias by using different splits of training and testing sets to evaluate overall performance. Dask enables the repeated training and testing of the models in an automated fashion.

Accuracy and **stability score** are two metrics to assess different machine learning pipelines' performance. They were calculated using following formulas:

$$StabilityScore = \frac{\sum_{n=1}^{\#runs} \bigcap rate}{\#runs} \qquad Classification\ accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

Results

Feature Selection	Classifier Model	Accuracy	Stability Score	Selected Features
RFE_SVM	Decision Tree	0.93	1.00	0.5-2Hz,4-6Hz, 8-10Hz, Peak frequency Broadband, theta/gamma ratio
Lasso score	Logistic Regression	0.90	N/A	Spectral Edge Density, 12-14Hz, 18-20Hz, Low Gamma, Zratio
RFE_SVM	Random Forest	0.89	0.78	0.5-2Hz,4-6Hz, 8-10Hz, Peak frequency Broadband, theta/gamma ratio
RFE_Logistic Regression	Linear Discrimination Analysis	0.87	1.00	0.5-2Hz, 8-10Hz, Peak frequency Broadband, theta/beta ratio, theta/gamma ratio
XGBoost	Logistic Regression	0.80	1.00	14-16Hz, 24-26Hz, 28-30Hz, Low Gamma, theta/gamma ratio

Table 1. Partial experimental results showed above. The **RFE feature selection and Decision tree classifier** had the best performance.

Summary and Future work

- We tested the pipeline with these combinations of feature selection methods and classification models, but it is designed in a modular fashion so that new methods can be easily included as long as they are made to be compatible with the sklearn API.
- The models were able to reach a classification performance of (accuracy: 93%, F1-score: 89%) on the first hour of the sample phenotyping dataset. This tool was also able to look at the optimal time window to observe phenotyping differences by feeding different time bins as well as their associated features into the models.
- The combinations of features identified provided a larger dynamic range than using any of the bins on their own. These meta-features, or spectral fingerprints, could be the key more robust and translatable EEG phenotypes.