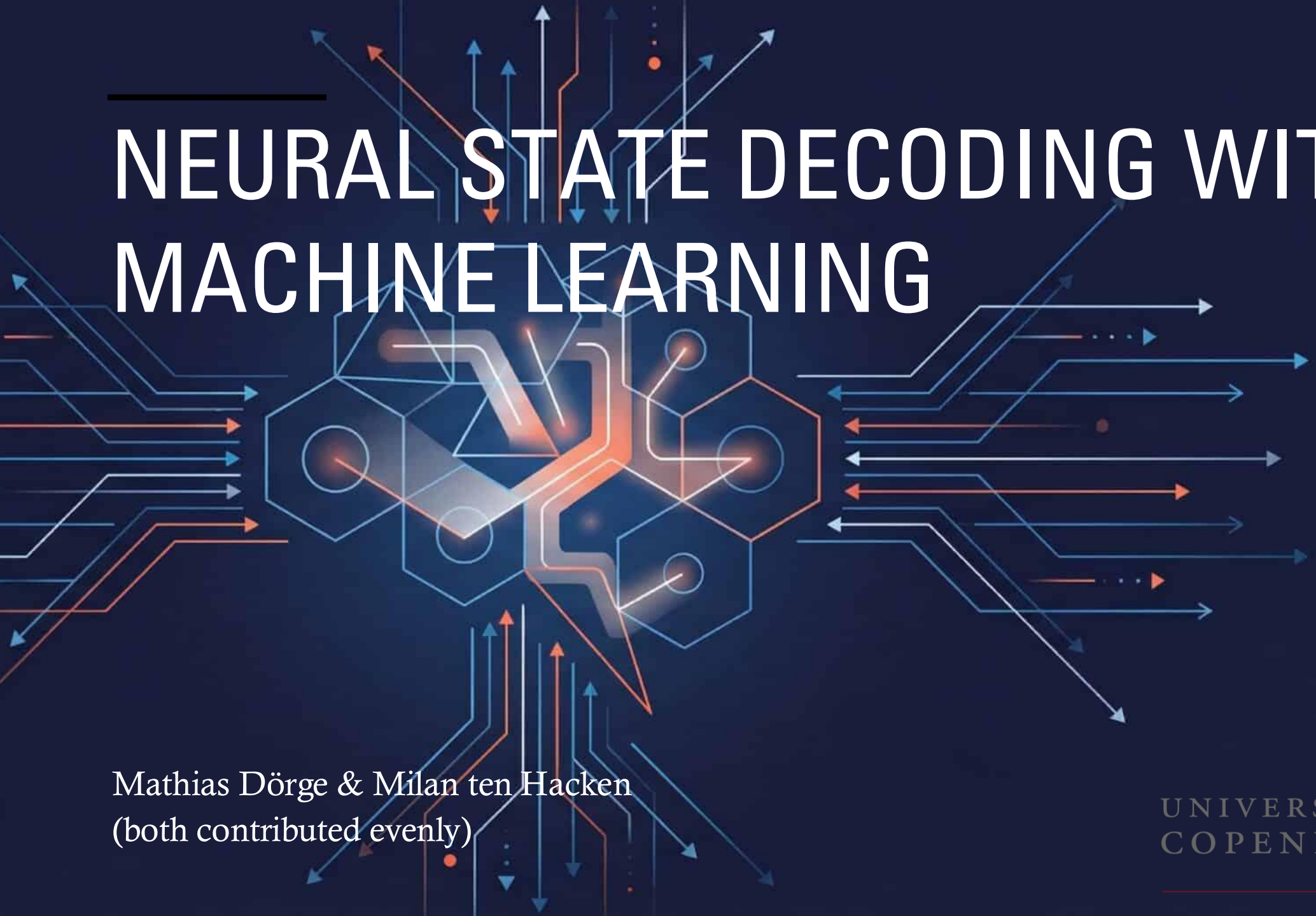
NEURAL STATE DECODING WITH MACHINE LEARNING



Mathias Dörge & Milan ten Hacken
(both contributed evenly)

UNIVERSITY OF COPENHAGEN

OVERVIEW

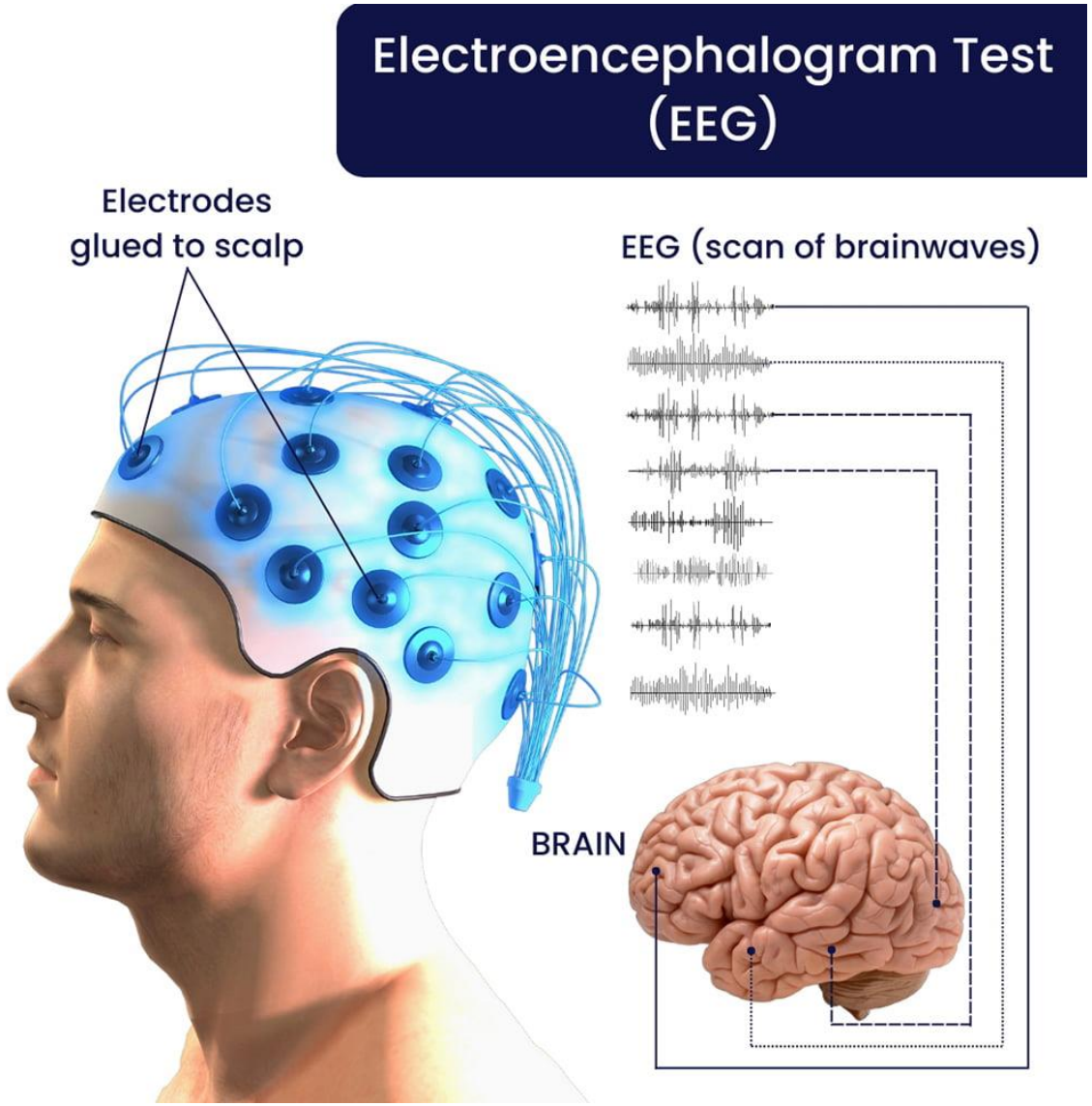
1. Motivation
2. The Data
    I. Visual inspection
    II. Clustering
    III. Feature importance
3. Sating the classification problem
4. Subject-specific model
5. Subject-general models
6. Evaluation

WHAT IS AN EEG?

- 19 standard electrodes placed on the scalp, measuring on different sections of the brain.
- 2 additional "lead electrodes".
- 1 Syncronization electrode. (removed for model training)

- Total of 21 channels.

- Measures changes in electrical signals with a certain frequency. (200 Hz and 1000 Hz in our data)



THE DATA

- Left/Right hand EEG measurements on 7 subjects:

example subject D

```
CLASubjectD1511253StLRHand.mat
EEG shape: (667600, 21)
Marker shape: (667600, 1)
Sampling frequency: 200
Channels: 21
```

Combined number of trials

```
Number of trials:

State 1: 4747
State 2: 4805
State 3: 4277
```

Total # of datapoints

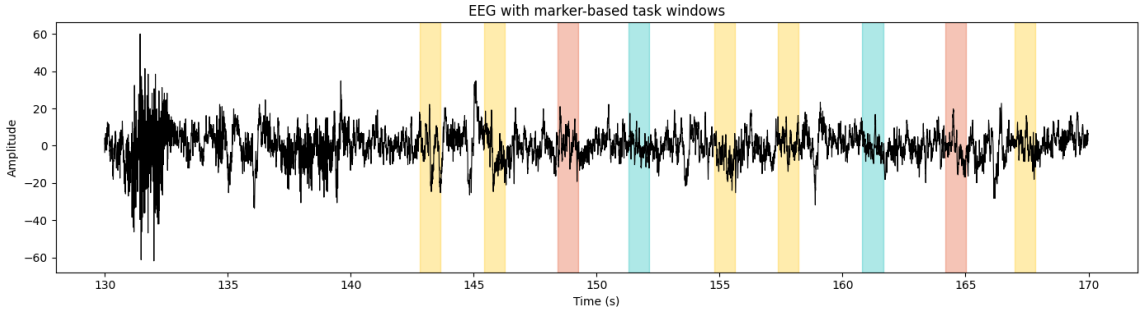
```
Dataset shape: (13829, 21, 170)
```

- 5 finger EEG measurements, High-freq and Low-freq data

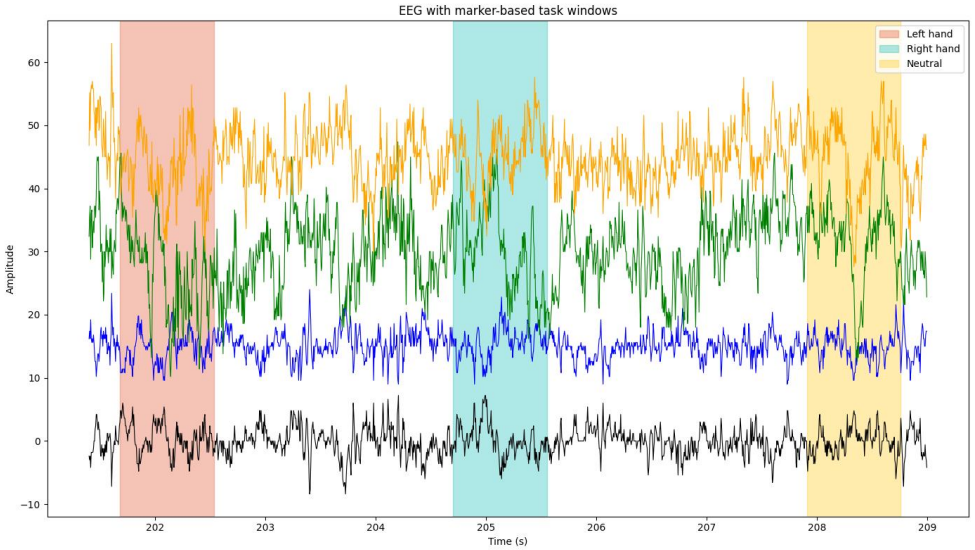
Paper: A large electroencephalographic motor imagery dataset for electroencephalographic brain computer interfaces
https://figshare.com/collections/_/3917698

VISUAL INSPECTION

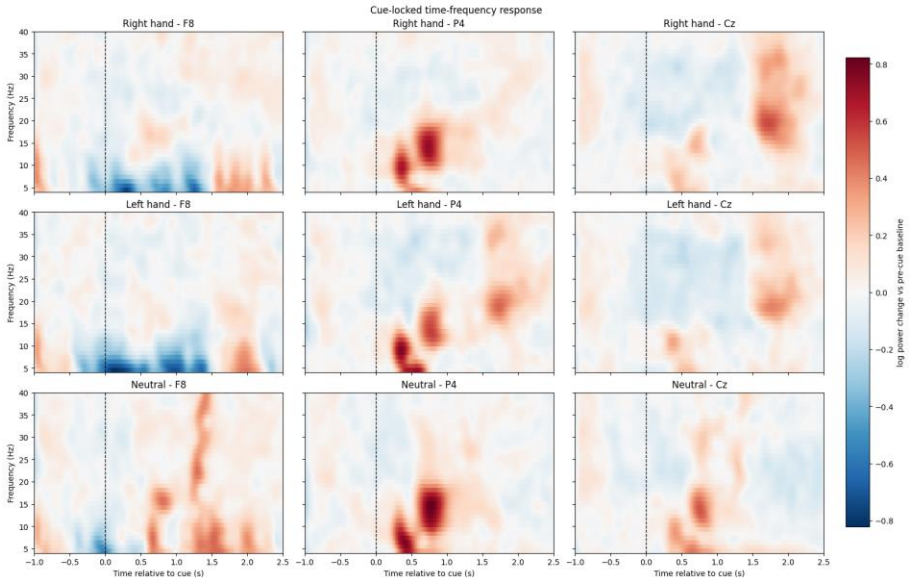
Plotting one electrode signal for one subject reveals… no structure



Closer up



Let's use the Fourier transform to inspect frequency dependent changes

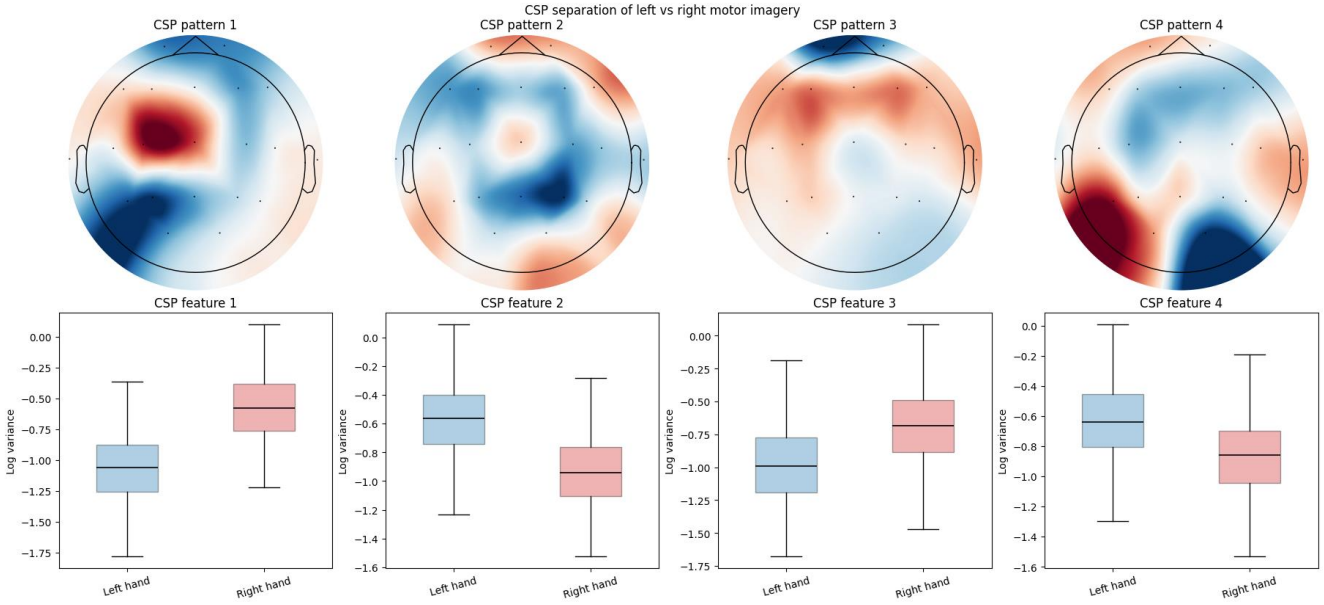


Better! But to our eye arbitrary differences between right/left/neutral

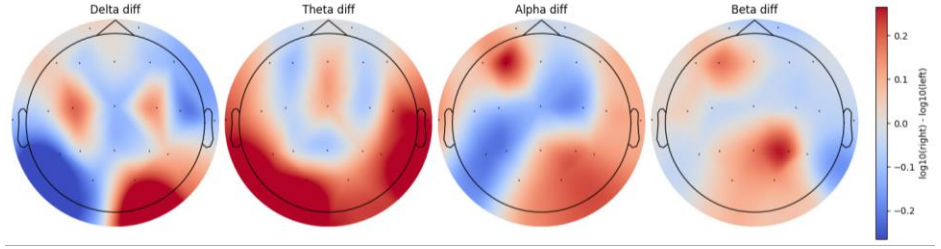
MNE for EEG analysis: https://mne.tools/stable/index.html

TOPOMAP AND CSP PATTERNS

- One way to extract features from the EEG's is using CSP from the mne library
- It finds the spatial filters that maximize variance, in this case spatial filters that maximizes left and right-hand signals.
  (or right vs. neutral, or left vs. neutral)



Literature says there are 4 distinct frequency bands
Delta (1-4Hz), Theta (4-8Hz), Alpha (8-12Hz), Beta (12-30Hz)



These are the average power differences of subject J

SUBJECT CLUSTERING (3 STATE)

Clustering using CSP features:
4 CSP filters per frequency band per marker (left/right/neutral) for each subject

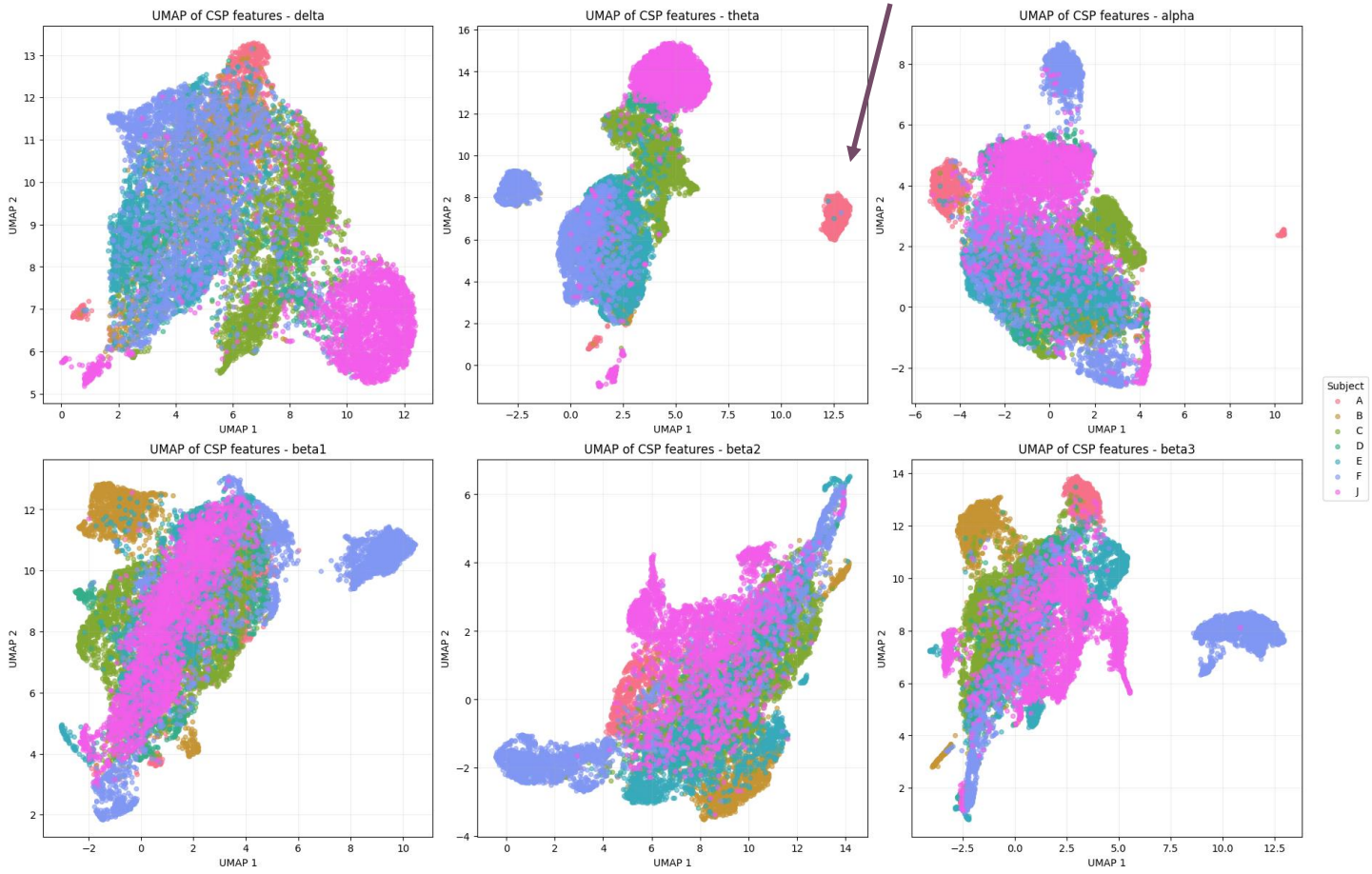
(4 filters x 3 markers x 6 bands = dim72)

Plotting into 2 dimensions with UMAP

That looks horrific…

Some subjects have very distinct brain patterns

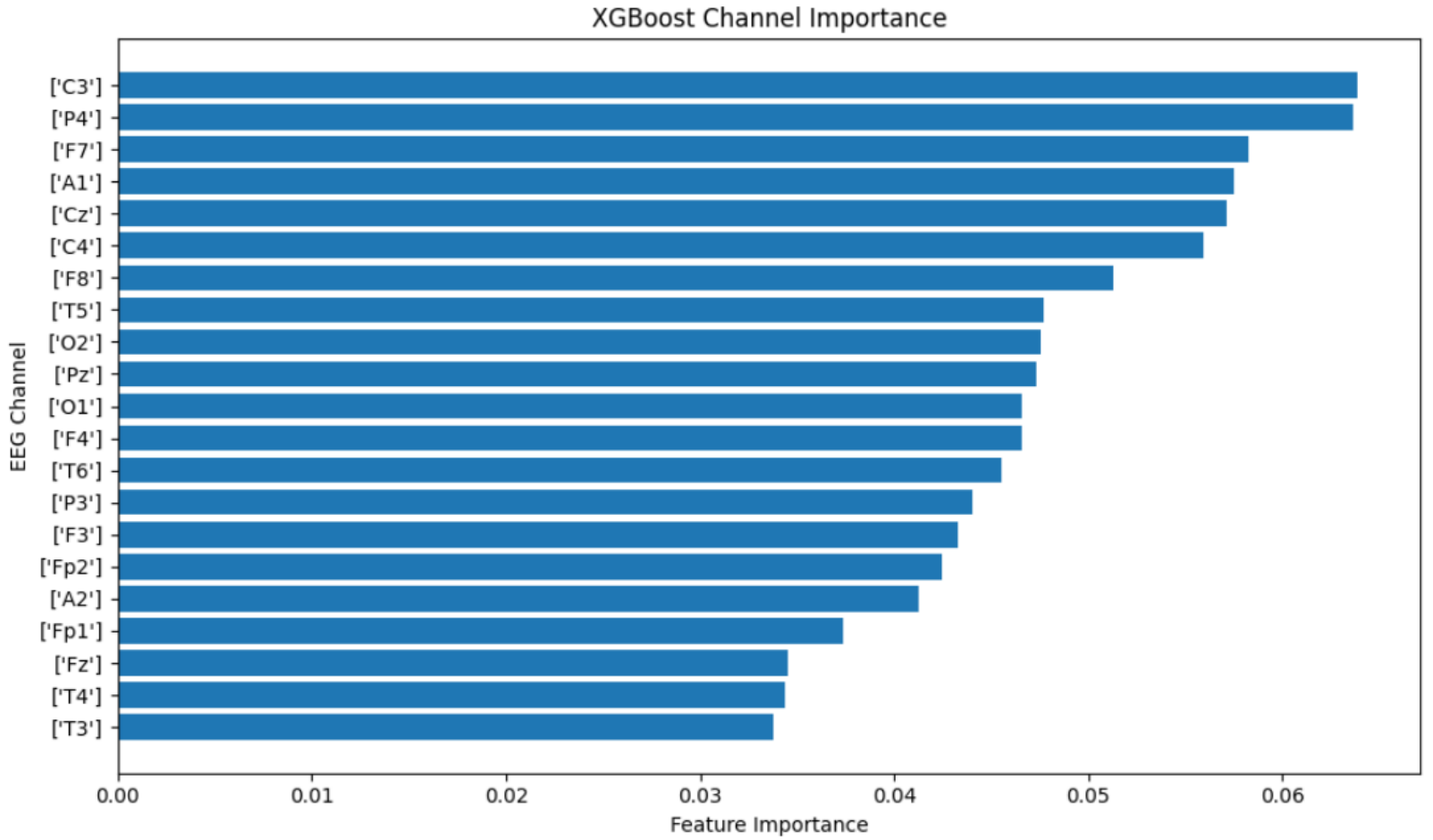
What was A thinking about?



ELECTRODE FEATURE IMPORTANCE

3 state: Left/Right/Idle

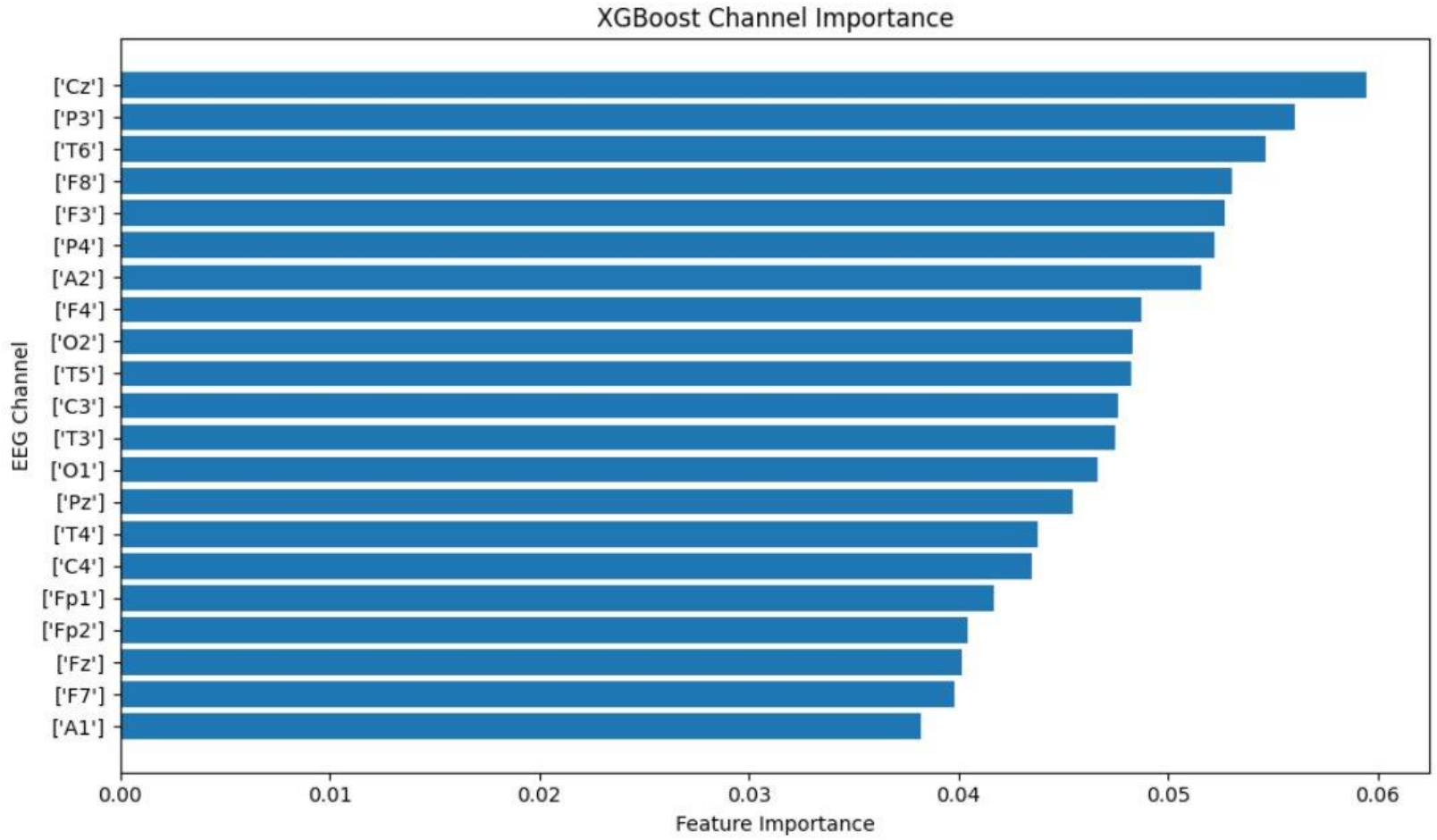
From literature we were expecting C3 and C4 to be important, as those areas of the brain should correspond to this motion.



ELECTRODE FEATURE IMPORTANCE

5 state: Five fingers low frequency

C3 and C4 significantly less important.

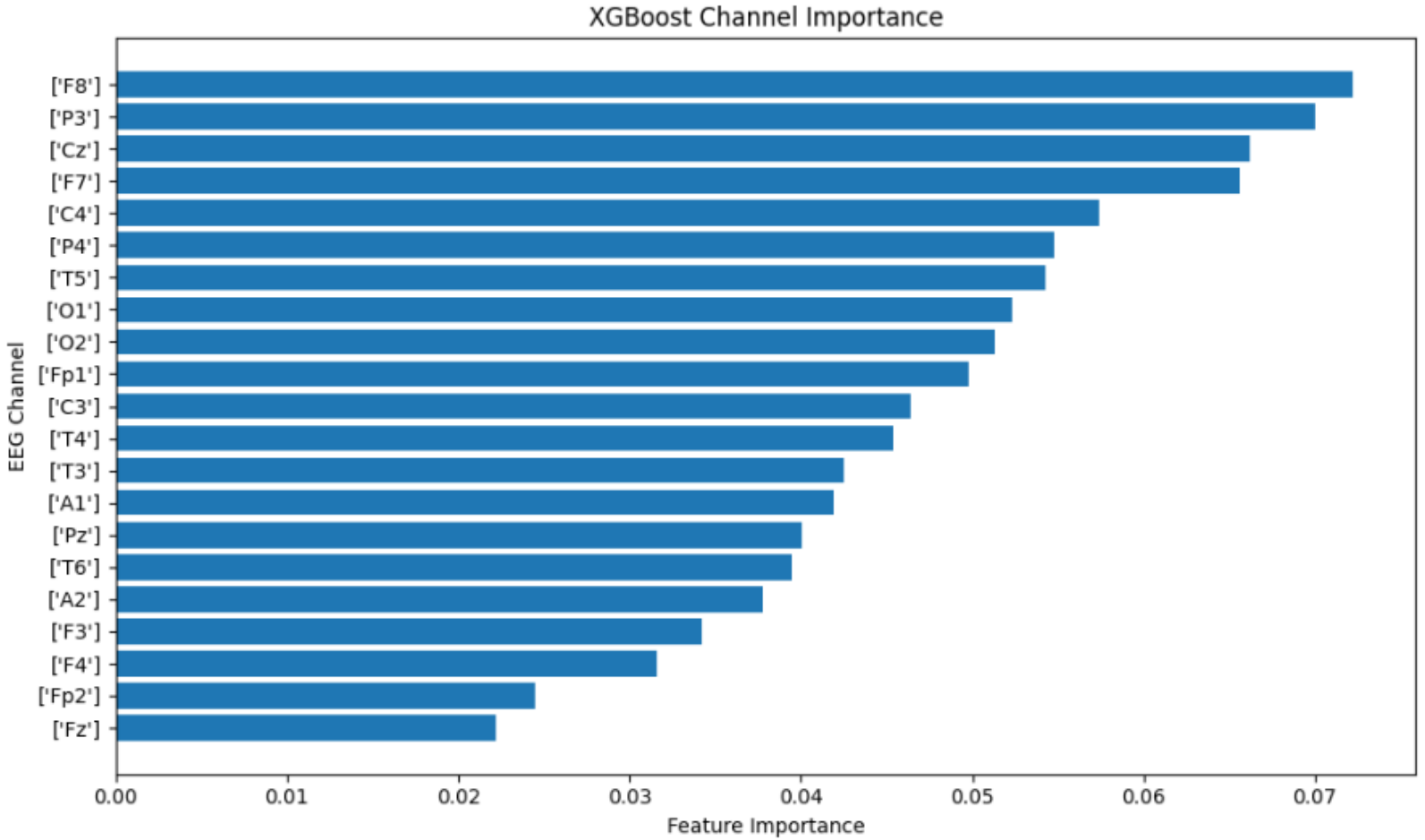


ELECTRODE FEATURE IMPORTANCE

5 state: Five fingers high frequency

Higher frequency data has different electrode feature importance.

Fewer very important features.



THE CLASSIFICATION PROBLEM

Classification

We want this one!

One subject prediction model

Idea: train and test on same subject

Use case: Mapping brain signals to motor tasks -> prosthetics

Execution: XGBoost, CSP or combination.

Difficulty: easy (at least for 3state data)

Subject invariant prediction model

Idea: train on all subjects, test using LOSO (Leave One Subject Out)

Use case: A general model, able to predict the neural state of any subject

Execution: XGBoost, CNN, adversarial training, clustering with CSP.

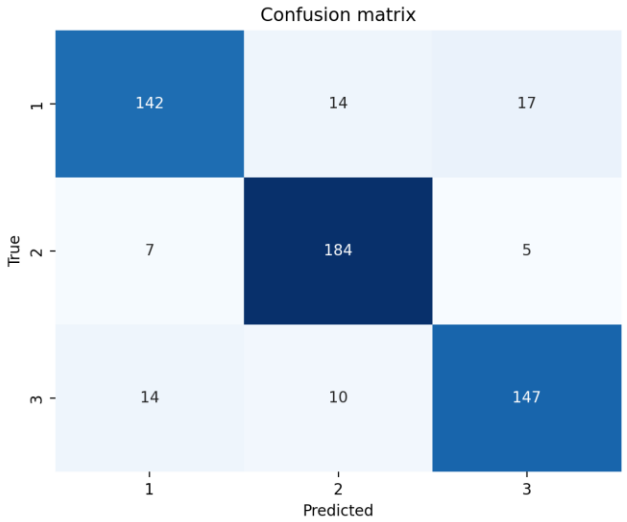
Difficulty: Very hard:(

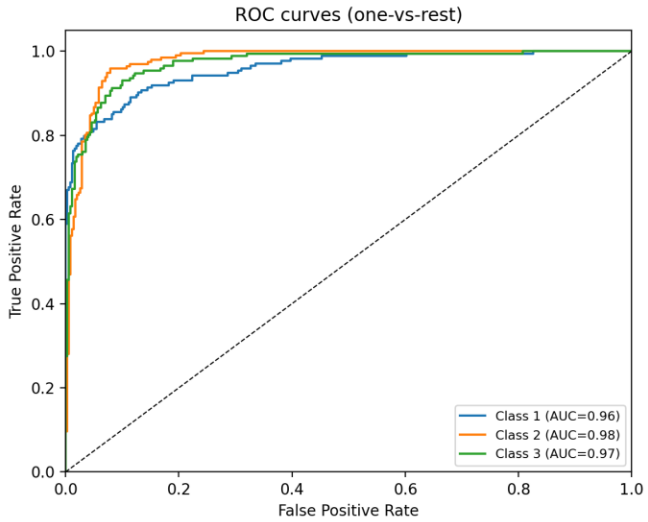
ONE SUBJECT PREDICTION MODEL (3 STATE)

- Using only subject J data to train and test (2700 trials total)
- Xgboost for simplicity and speed
- Hyperparameter optimization (random search)

Classification report:

	precision	recall	f1-score	support
1	0.87	0.82	0.85	173
2	0.88	0.94	0.91	196
3	0.87	0.86	0.86	171
accuracy			0.88	540
macro avg	0.88	0.87	0.87	540
weighted avg	0.88	0.88	0.88	540



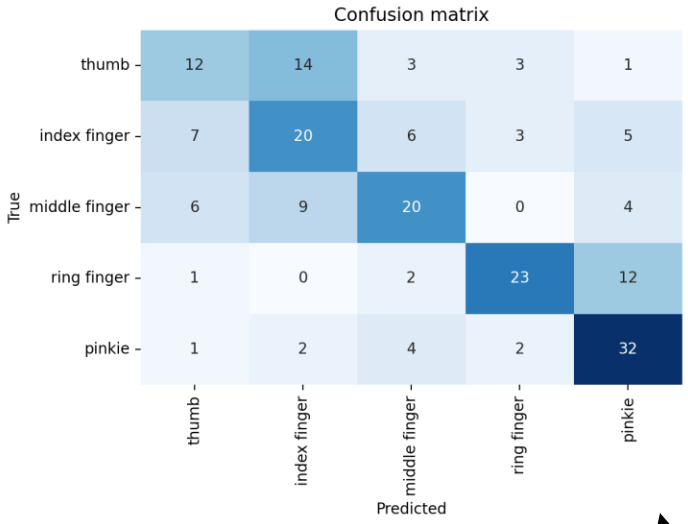


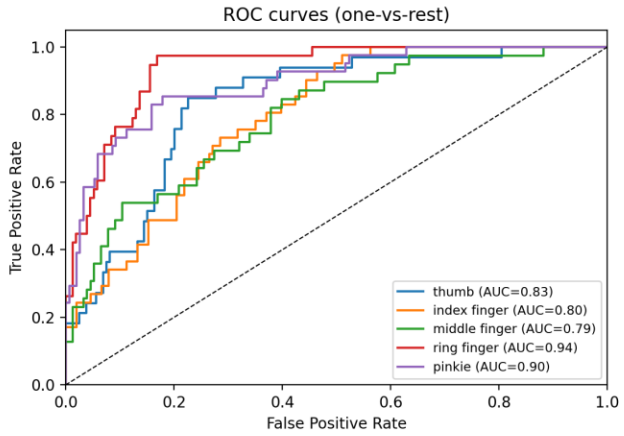
It is easy and quick and gives near 90% test accuracy!

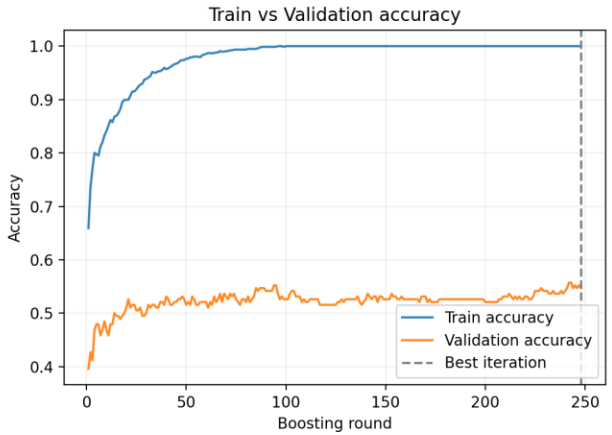
Params: eta=0.05, Eval metric: 'mlogloss', max_depth=3

ONE SUBJECT PREDICTION MODEL (5 STATE)

Subject C: 900 trials







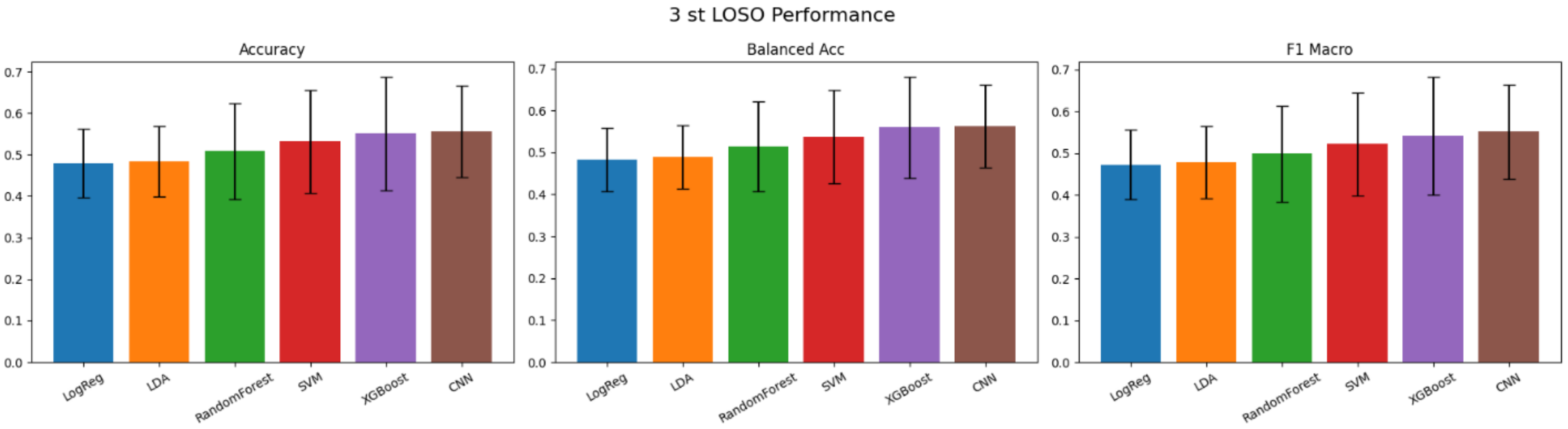
Pinkie does really well!

accuracy	0.56

The drop in performance is expected: distinguishing between 5 states is much harder

Params: eta=0.05, Eval metric: 'mlogloss', max_depth=3

MODEL COMPARISON 3 STATE



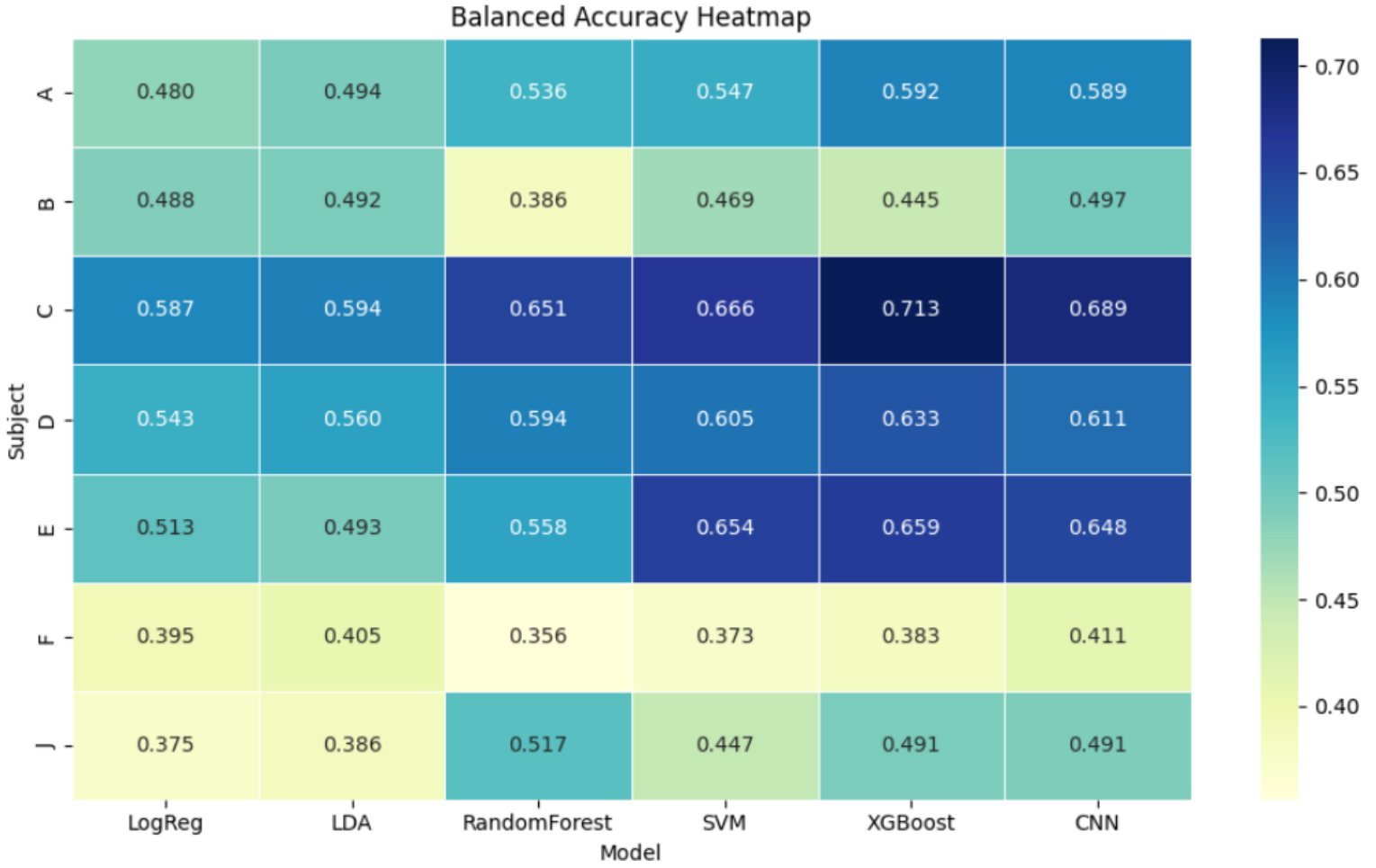
Best performing models
XGBoost and CNN

COMPARING NON-OPTIMIZED STANDARD MODELS

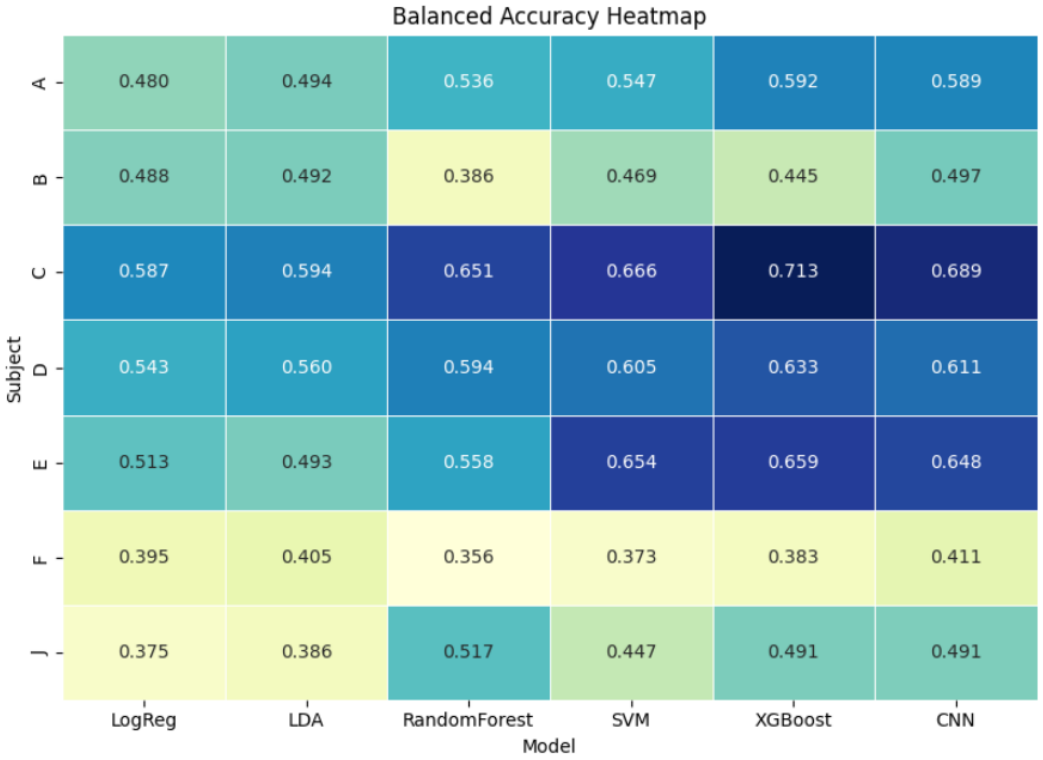
Random for 3 state: 0,33

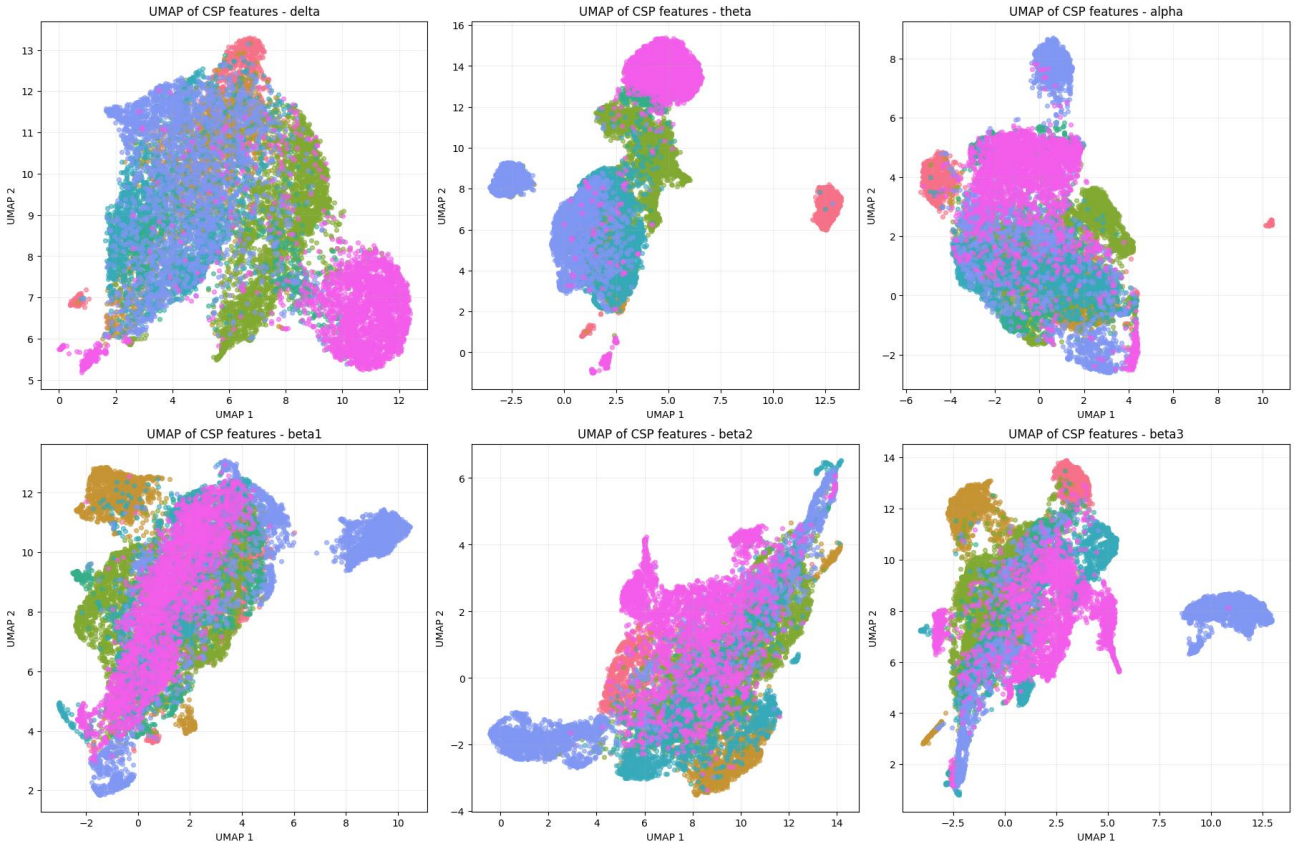
Notice subject C and F

C: Generally good
F: Generally bad

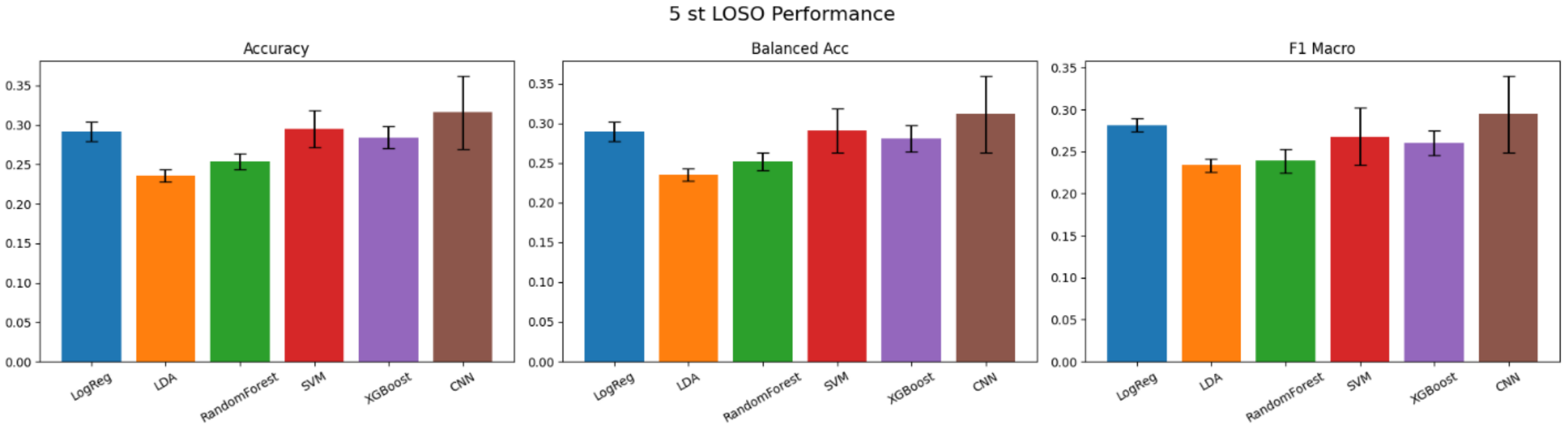


Subject F: Blue
Subject C: Green





MODEL COMPARISON 5 STATE



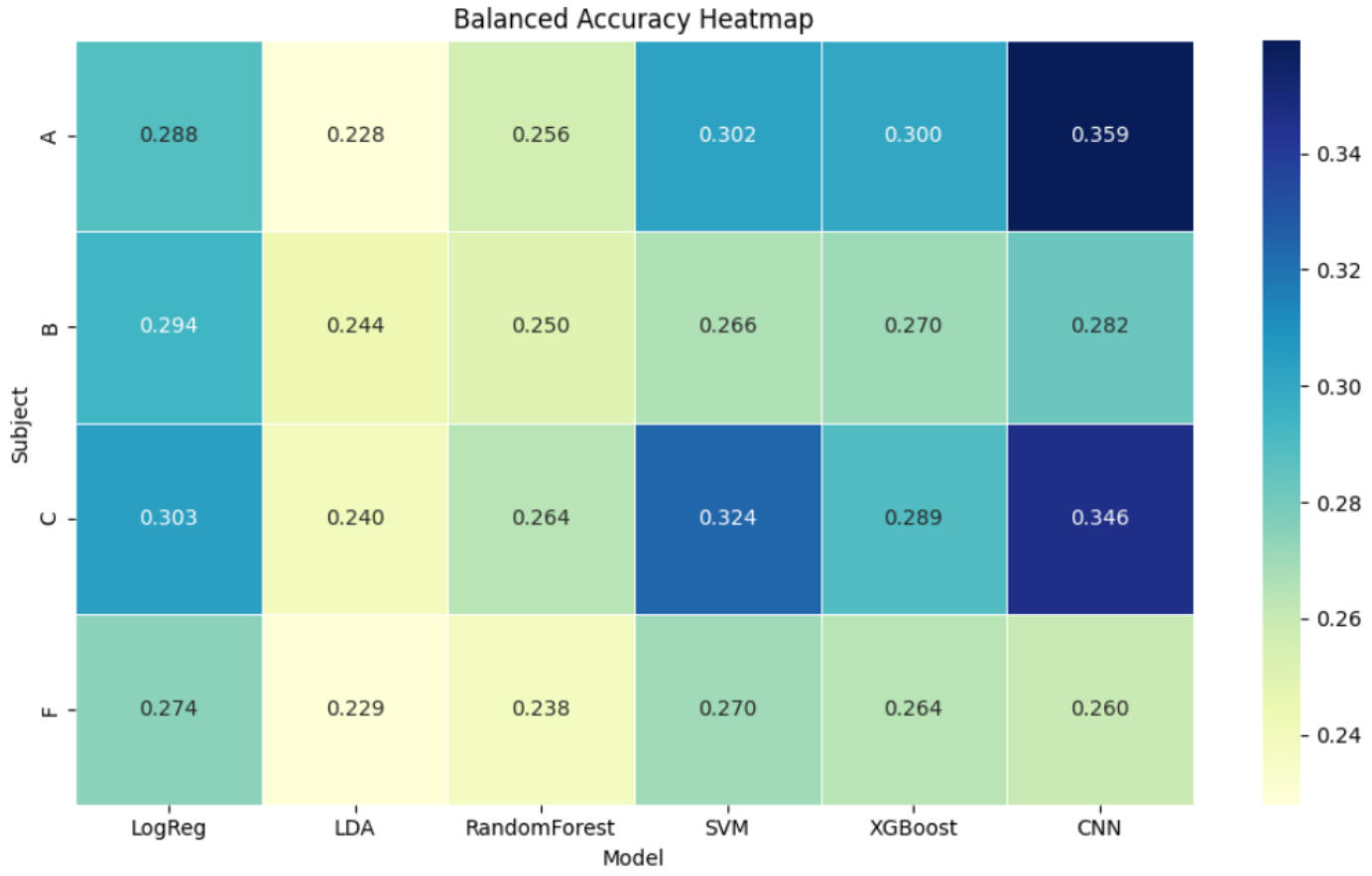
Once again best performing:
CNN
This time also
LogReg (But SVM and XGBoost close by)

COMPARING NON-OPTIMIZED STANDARD MODELS

Random for 5 state: 0,20

Small variations in accuracy across models and subjects.

CNN does best in general and LDA is just about random



OPTIMIZED PERFORMANCE

Using optuna to do hyper parameter optimisation for CNN and XGBoost in the 3 st classification.

XGBoost

Improvement: 0,713 --> 0,720

CNN

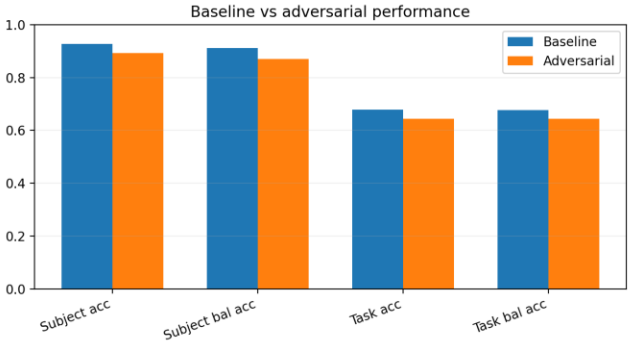
Improvement: 0,689 -->0,722

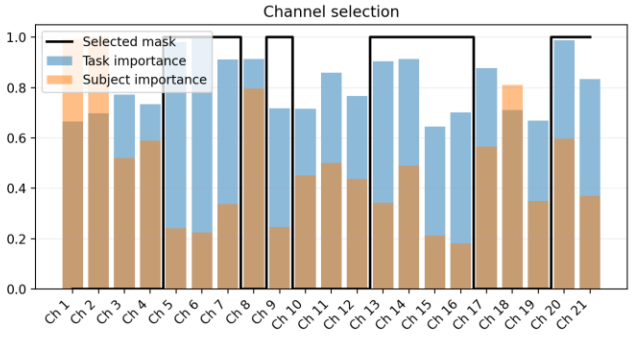
Small improvements, from hyper parameters, but large impact from subject invariance.

ADVERSARIAL TRAINING (3 STATE)

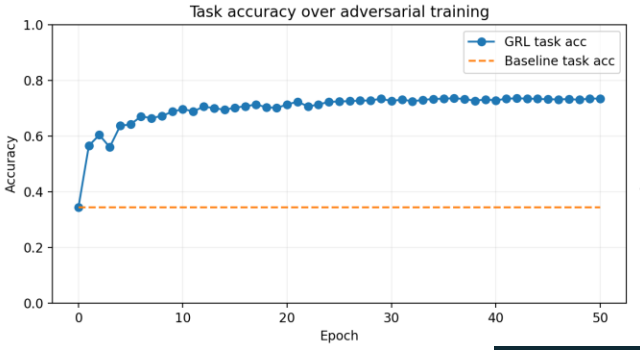
The goal: minimize subject features, while maximizing prediction on the marker (left/right/neutral)

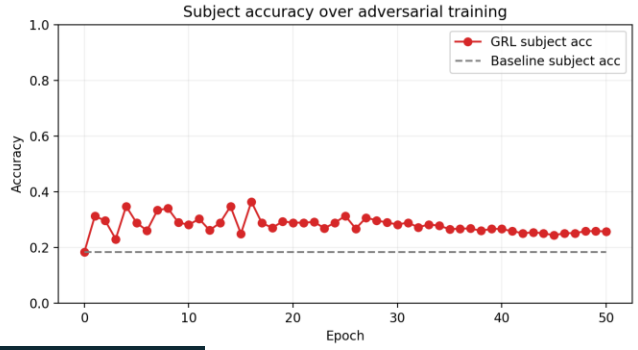
Running two classifiers reveals the channel importance for predicting subjects and tasks. Some channels give more information away on the subject.





Gradient Reversal Learning (GRL) + adversarial training allows us to punish subject prediction and update the feature extractor over multiple epochs.





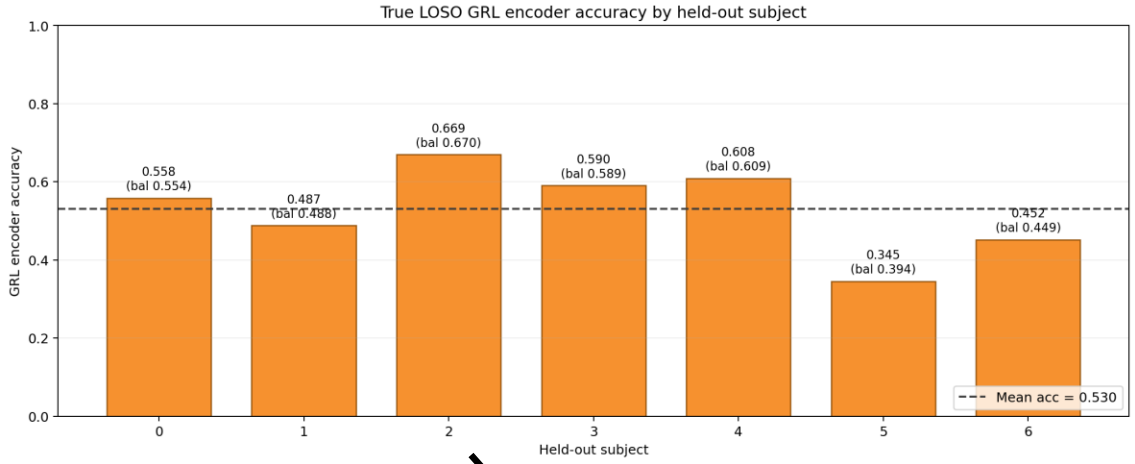

```
Final evaluation after adversarial training:
Task accuracy: 0.7321
Task balanced accuracy: 0.7328
Subject accuracy: 0.2440
Subject balanced accuracy: 0.1877
```

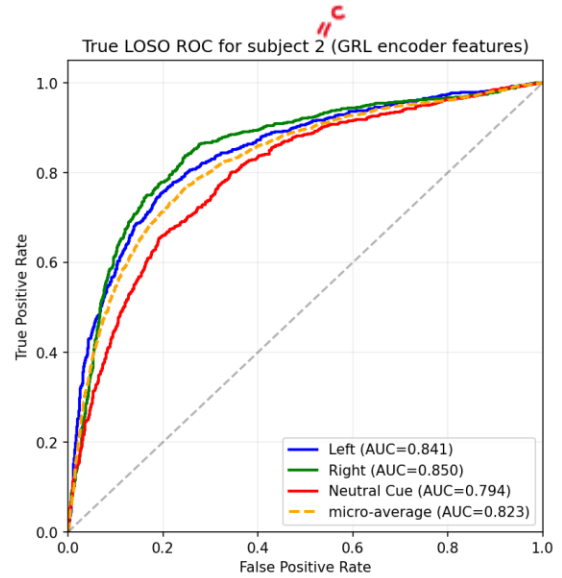
Note! This is not using LOSO yet

RESULTS OF THE TRAINED ENCODER

- The adversarial trained encoder still did quite poorly when leaving one subject out… And gets 4th place in all our models.

Note! Subject C has always performed the best in our models, why is that? Remember the clusters: Subject C is the most centred subject and shares the most features with all others.





CONCLUSION

- The challenges in this project:
  - Pre-processing the data (marker extraction, data handling)
  - Extracting the useful features (Fourier Transform, CSP spatial filters, bandpass filter?) Should we stay in time domain with a CNN or go all in on spatial features?
  - Subject specific features, annoying because the model will train to recognize subjects. Adversarial training got this down but without noticable accuracy gain.
  - Limited dataset: 7 subjects. LOSO will never be great with few subjects.
- Improvements provided more time:
  - Using an EEG foundation model (from braindecode) and train the last layers with our data.
  - Using all high frequency 5 finger data and optimize further (smarter methods or more computing power).
  - Merge finger data and R/L hand data to make one generalized model.
  - Investigate brain pattern clustering and researching subject differences (privacy issues).

OWN EVALUATION

- 3 state

Models did fairly well for some subjects and poorly for others.

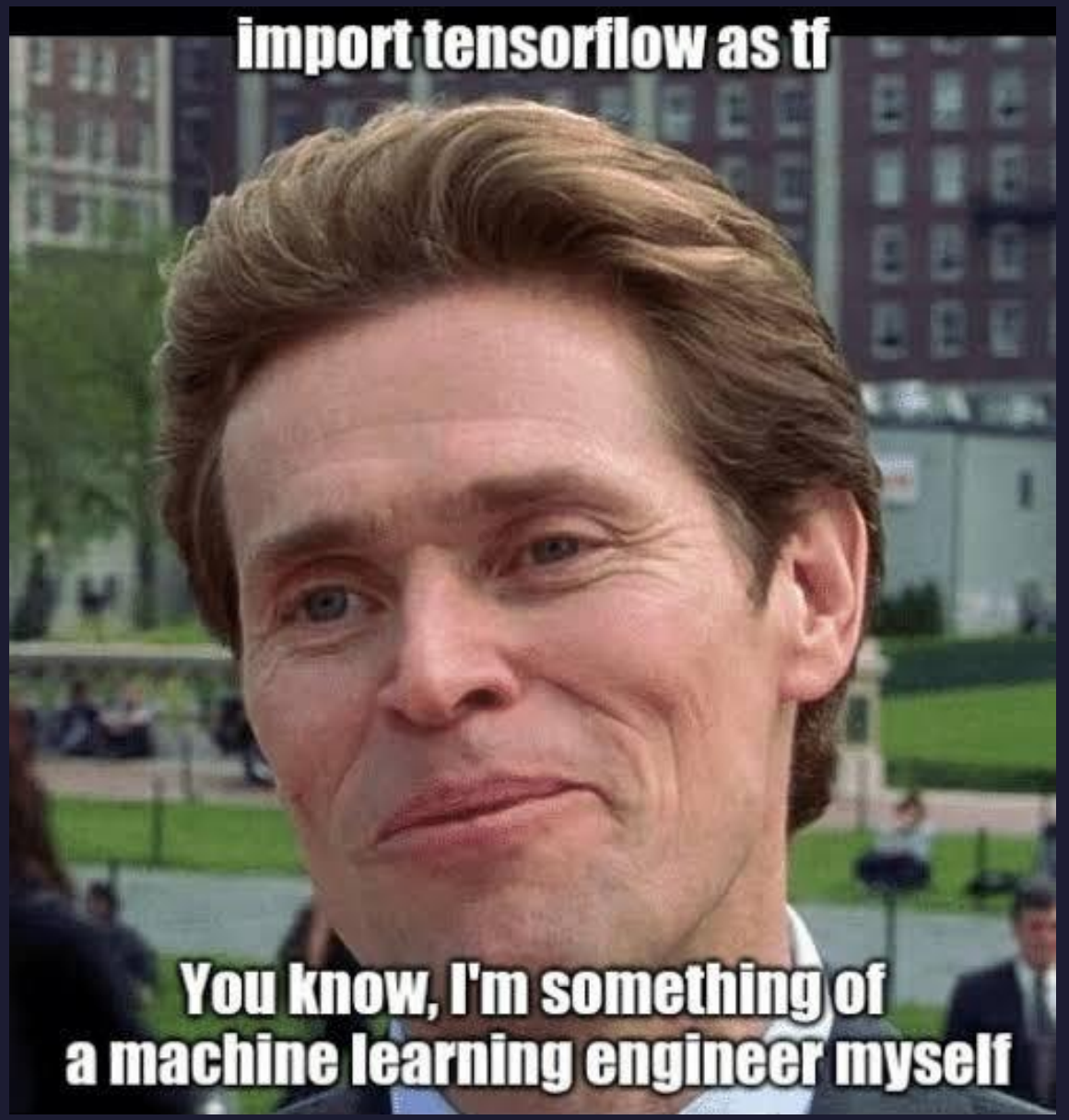
- 5 state

Models did significantly worse, but with more states a lower acc was expected.

Compared to litterature, a fair performance.

- Room for improvement (ideas from last slide)

THANK YOU!



- By Milan Ten Hacken and Mathias Dörge

WORKLOAD

- All participants contributed evenly

APPENDIX

Anything we did, saved, and thought was useful in the project are either in the presentation or these slides.

Those include: figures from old models, hyper parameters for the models

We have tried to refrain from duplicating stuff from the presentation in the appendix.

PROJECT STRUCTURE

1) Find suitable dataset for machinelearning

We found a study with a few subjects, but hours worth of data gathered from each subject, specifically gathered for the purpose of machinelearning.

Collection - A large electroencephalographic motor imagery dataset for electroencephalographic brain computer interfaces - Scientific Data - Figshare

2) Read the paper and other sources to gain (some) domain knowledge and get an idea of the data gathering and the uses for such data.

3) Load in the data and inspect how the data was structured.

4) Extract the trials (what we want to classify) and inspect feature importance.

5) Try out standard methods for EEG data (CSP, bandpassfilter)

6) Cluster CSP for different subjects to try and see how similar subjects are.

7) Train and test on the same subject, to see how well models train on a subject, where the trials are actually similar.

8) Let standard not optimized models train on all but 1 subject and test on the one, to get a baseline idea of how different models perform and see how leaving different subjects out affect performance.

9) Optimize CNN and XGBoost (best performing models for 3 st) using optuna. (CNN to keep timeseries structure)

10) Do adversarial training on the 3 st data to avoid subject recognition.

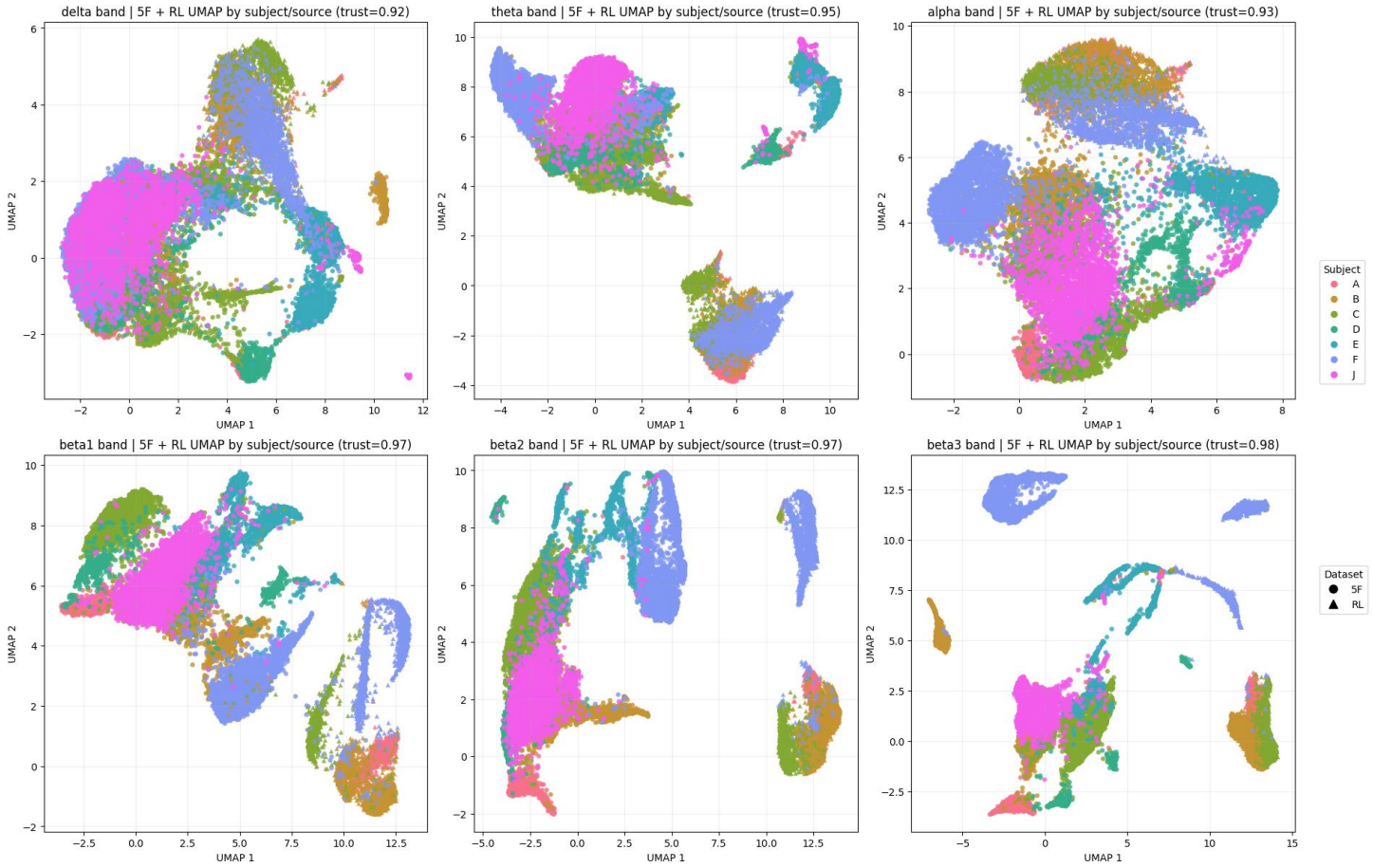
BEST HYPER PARAMETERS XGBOOST AND CNN FOR LOSO WITH SUBJECT C

- XGBoost
- Trials run (30) with optuna
- Accuracy: 0.720389023966655
- Params: {'n_estimators': 500, 'max_depth': 5, 'learning_rate': 0.021791501837346117, 'subsample': 0.5659726234738243, 'colsample_bytree': 0.5806588112073384, 'gamma': 2.129846022037345, 'min_child_weight': 9}

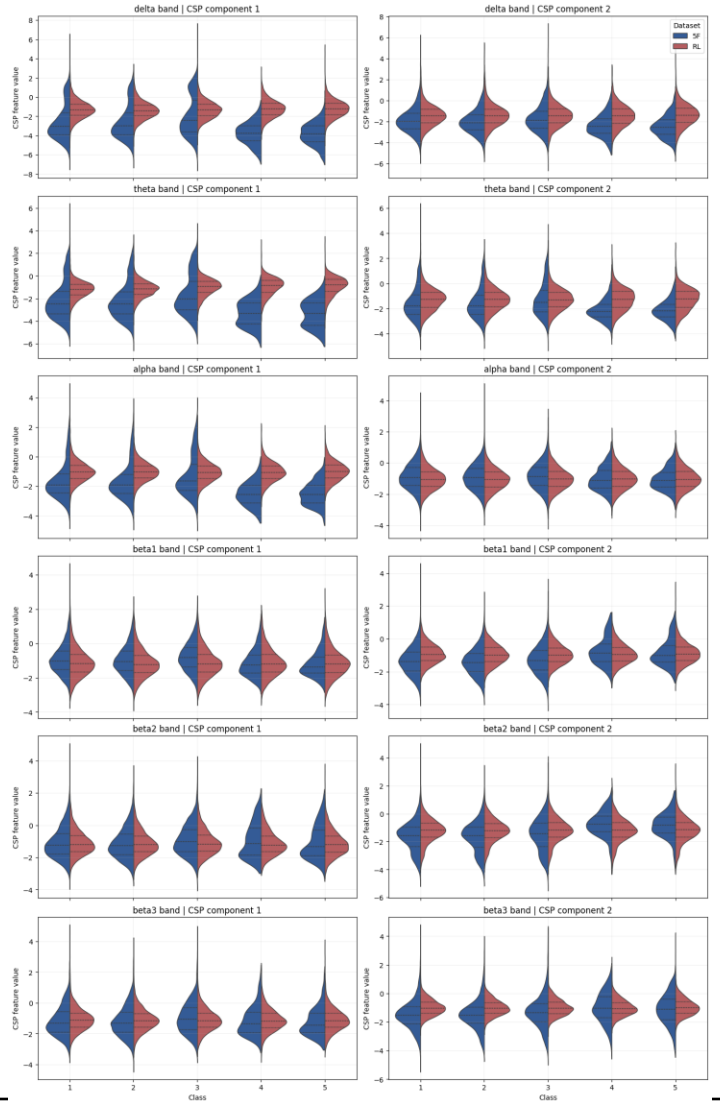
- CNN
- Trials run (30) with optuna
- Accuracy: 0.7224730809308788
- {'filters1': 64, 'filters2': 128, 'kernel_size': 7, 'dense': 128, 'dropout': 0.5118079701762697, 'lr': 0.0023267689816288 16}

- These were the best XGBoost and CNN for LOSO with subject C

3STATE + 5 STATE DATA



Clustering with CSP patterns



CSP feature value comparison

HYPERPARAMETERS

Architecture adversarial trainer:

```
Self.encoder = nn.Sequential(
nn.Linear(input_dim, 512),
nn.ReLU(),
nn.Dropout(0.3),
nn.Linear(512, 128),
nn.ReLU(),

self.task_head = nn.Sequential(
nn.Linear(128, 64),
nn.ReLU(),
nn.Linear(64, n_task_classes),

self.subject_head = nn.Sequential(
GradientReversal(grl_lambda),
nn.Linear(128, 64),
nn.ReLU(),                          num_epochs = 50
nn.Linear(64, n_subject_classes),   subject_loss_weight = 0.5
```