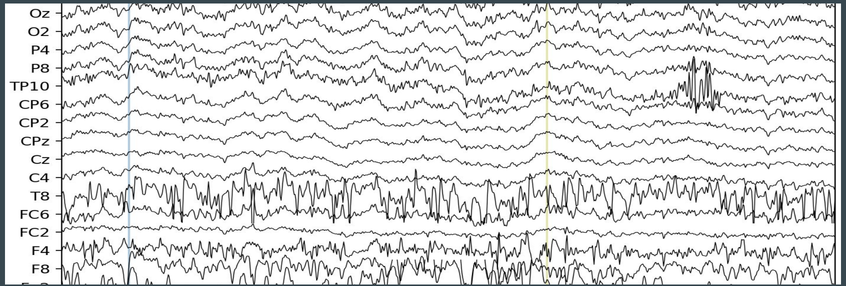
# Analysis of Electroencephalogram (EEG) Data with Machine Learning

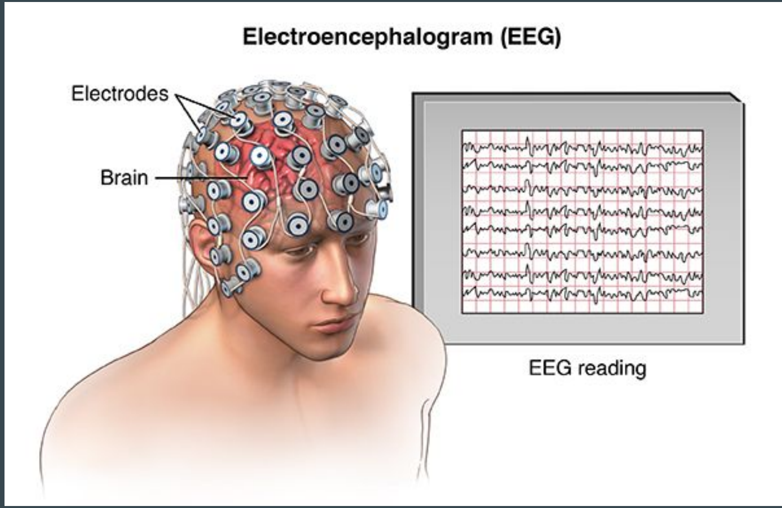
CMPT 496 Capstone

Jesse Emery, Nhi Phan, Isra Jime
Supervisor: Dr. Dana Cobzas & Dr. Cameron Hassall

# Background

- EEG – Studying electrical brain activity
  - Measures electrical signal through electrodes placed on the scalp
- Recorded on electrogram
  - Channels for each node
- EEG changes during event
  - Ex. motor response, reward

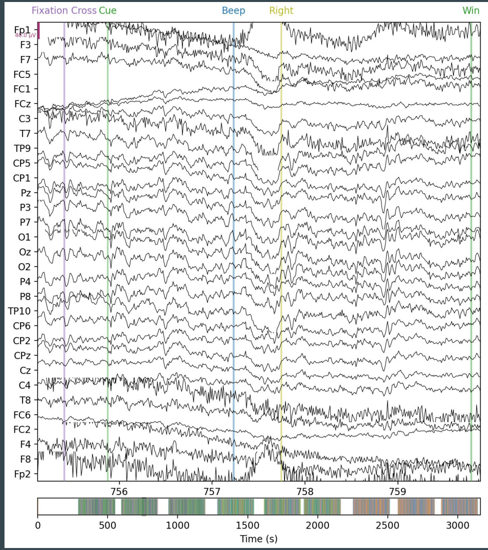



Taken from https://www.brightbraincentre.co.uk/electroencephalogram-eeg-brainwaves/

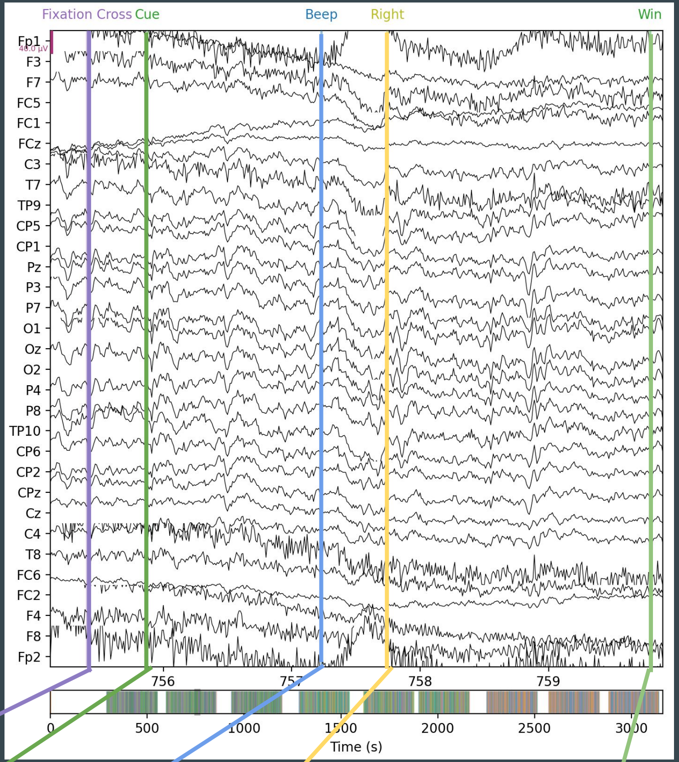
# Process

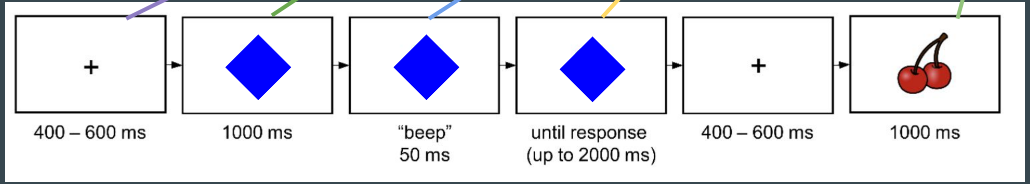
**Input Data** → Classifiers → Results



# Data

- Past experiment from Dr. Cameron Hassall
- 12 Participants
  - Focused on 1 participant
- 31 electrode nodes (channels)
  - Sampled at 1000 Hz
  - Measured in μV
- Task:
  - Stimulus: Color + Shape
  - Motor: Left or right
  - Outcome: Win or loss
- 427 tasks over 50 minutes



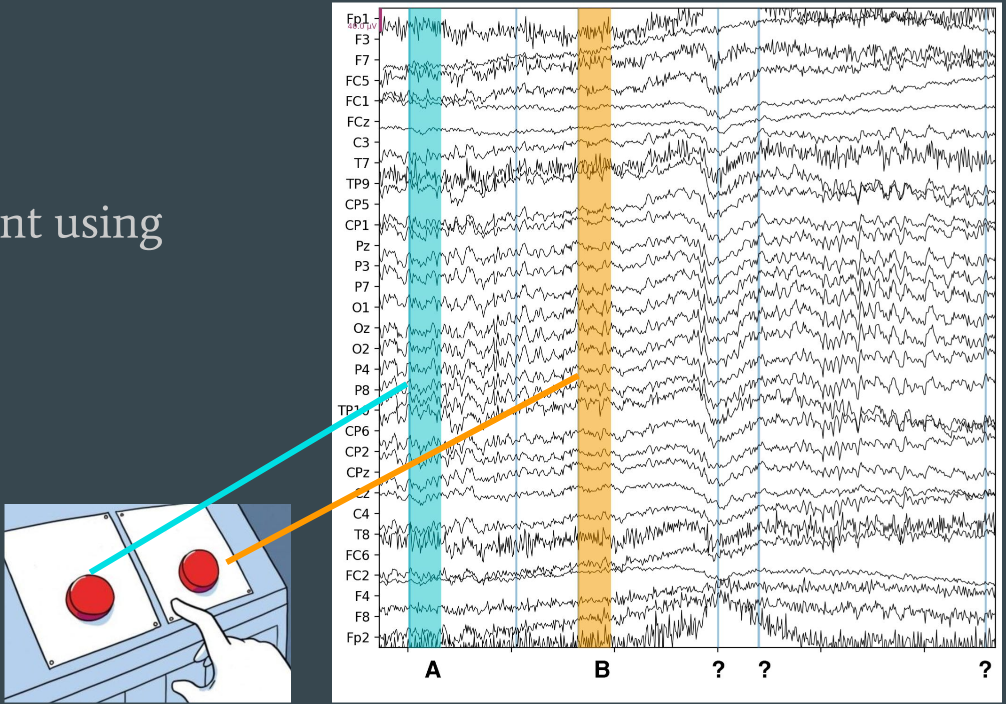



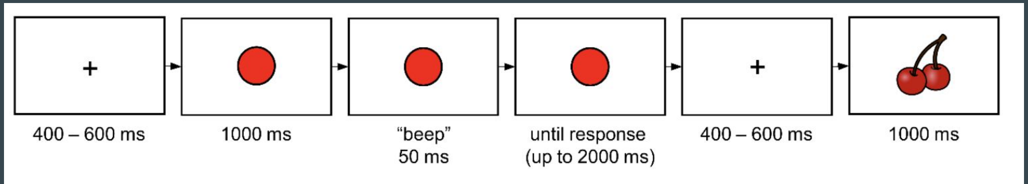
# Problem

Can we accurately classify an EEG event using machine learning on past events?

Event types:

- Stimulus: Color + Shape
- Motor: Left or right
- Outcome: Win or loss



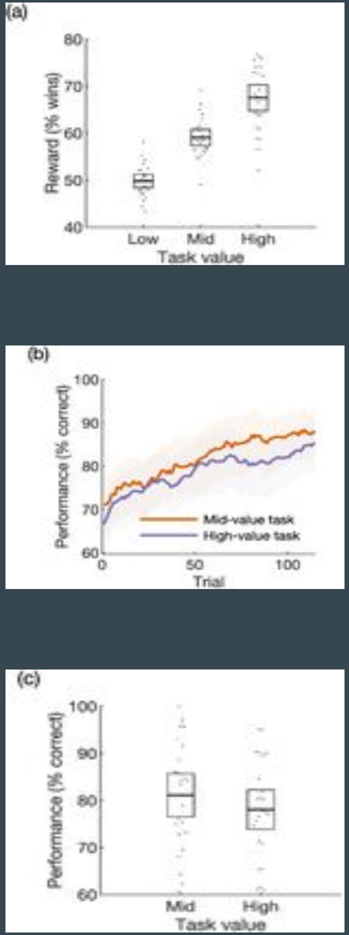
# Related Works

Dr. Cameron Hassall - “Task-Level Value Affects Trial-Level Reward Processing”

- Anterior cingulate cortex is involved in decision making

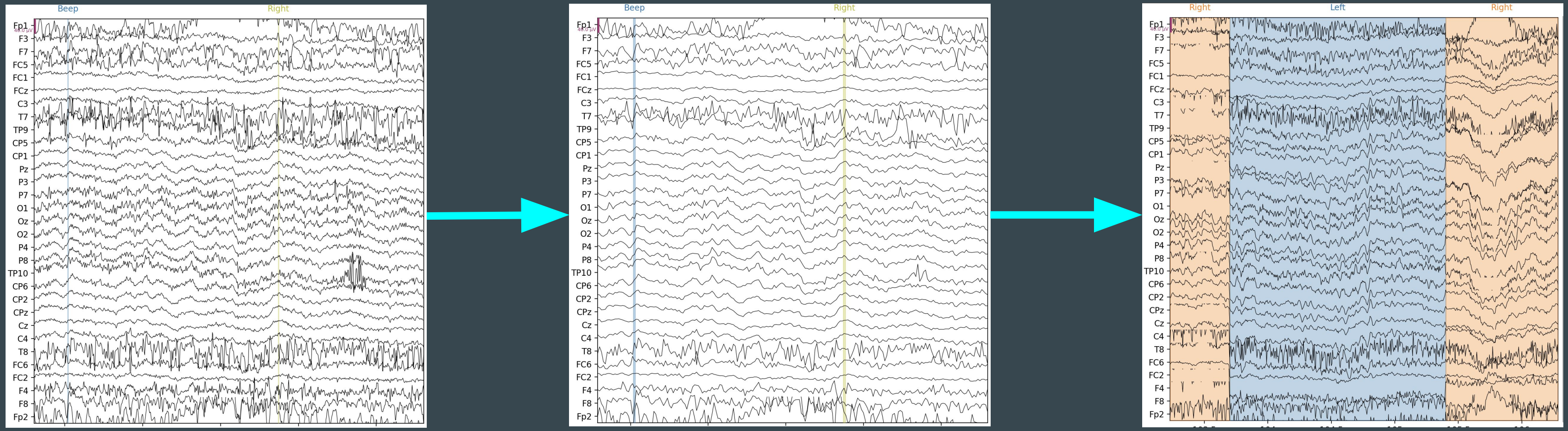
Data Analysis:

- Used MATLAB 2022a + EEG library
- Methods
  - Removed eye-related artifacts using independent component analysis
  - Difference wave approach - subtracting mean loss ERP from mean win ERP
  - Electrode location + time window examination
  - Computational modelling to determine if different strategy was used
- Findings:
  - “ ACC evaluates outcomes based on value of tasks rather than cues in isolation”.

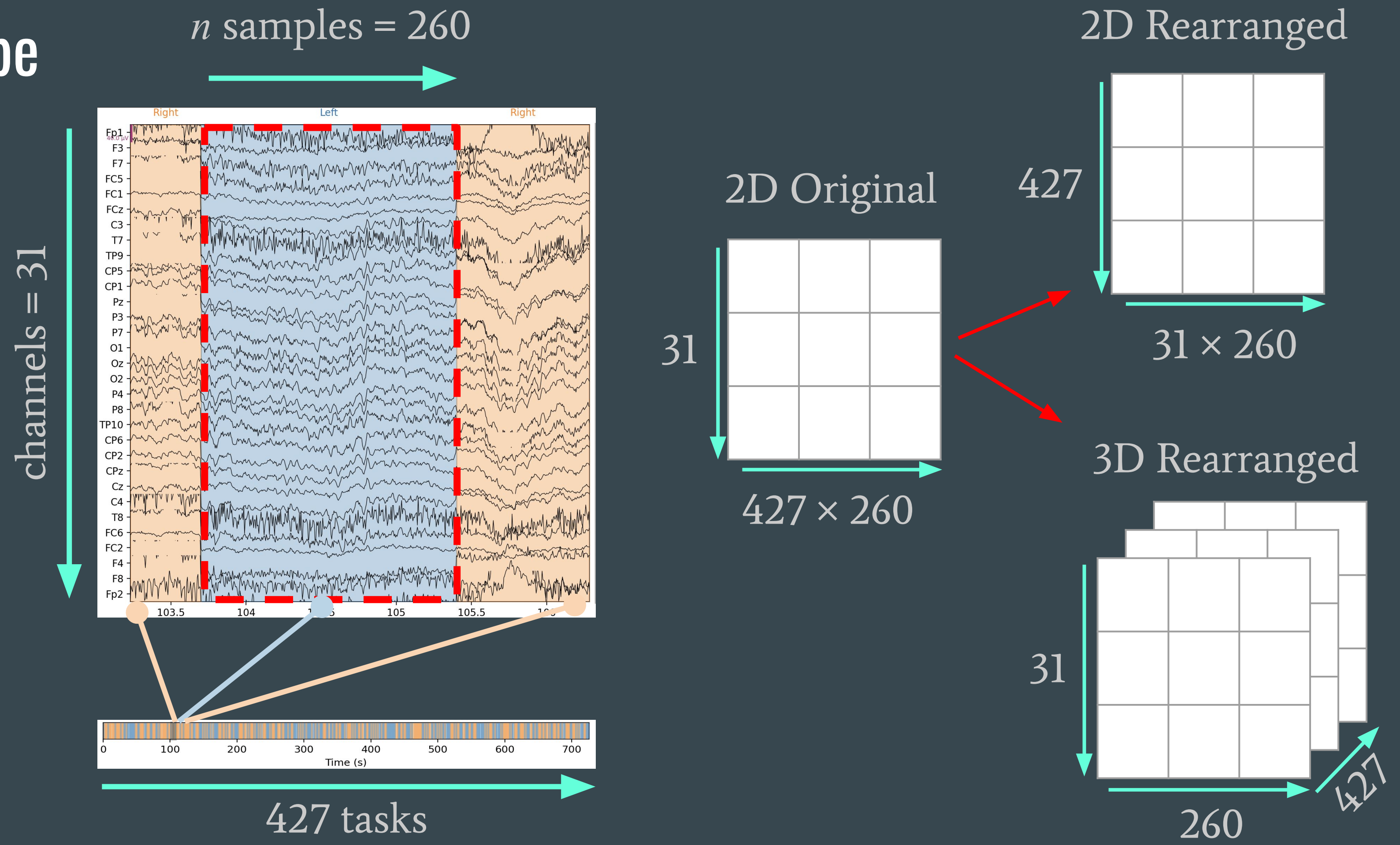


# Preprocessing

1. Map event IDs to types
2. Resample
3. Crop motor events
4. Recombine
5. Reshape

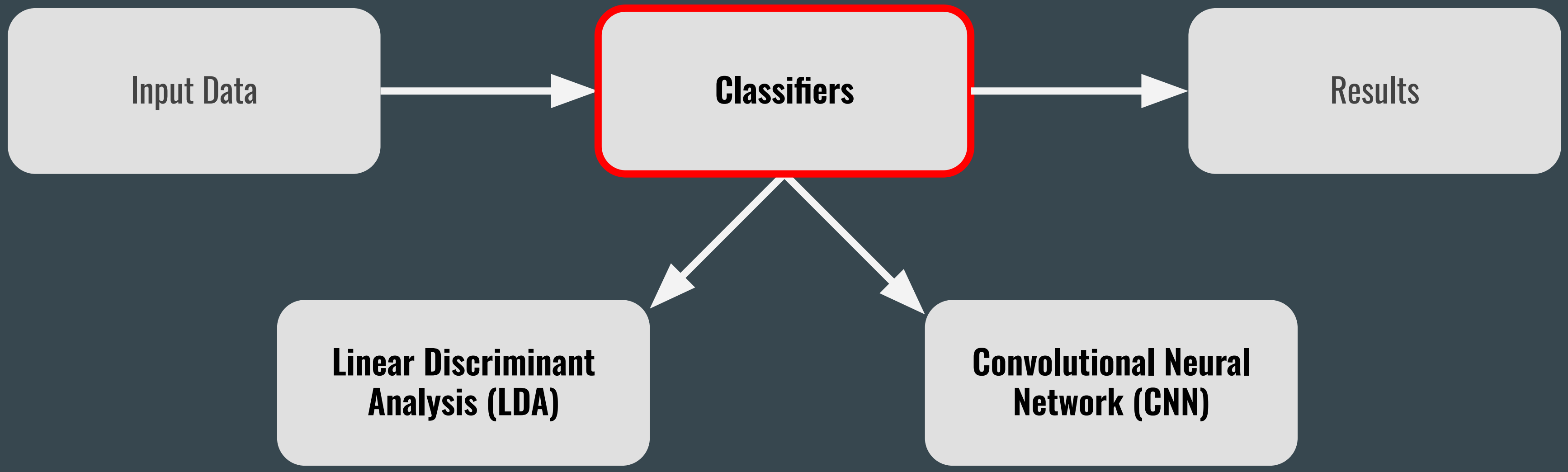


# Reshape

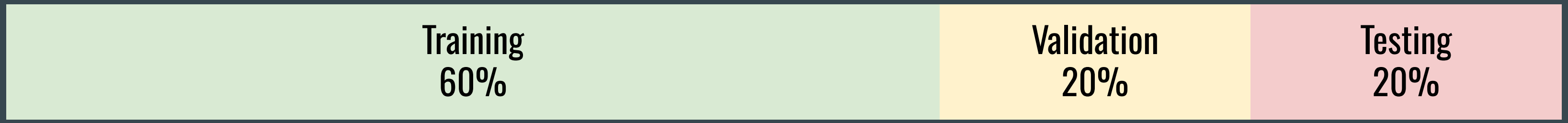


n samples = 260
channels = 31
Right
Left
Right
Time (s)
427 tasks
2D Original
31
427 × 260
2D Rearranged
427
31 × 260
3D Rearranged
31
260
427

# Classifier Types / Evaluation




Split data and labels into 3 parts



# Analysis - SciKit

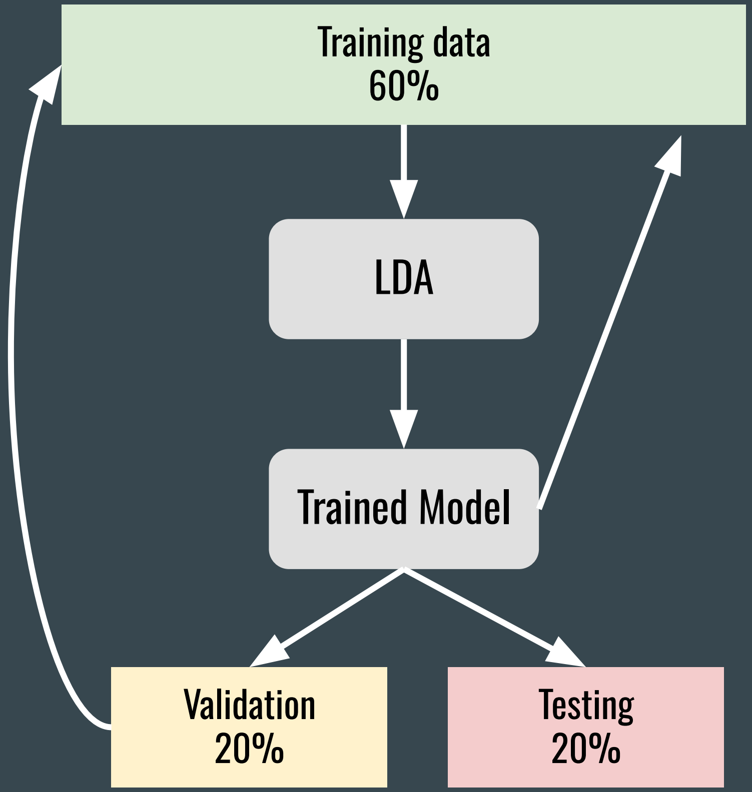
## Linear Discriminant Analysis (LDA)

$$\delta(x) = x * ( \sigma^2 * (\mu_0-\mu_1) - 2 * \sigma^2 * (\mu_0{}^2-\mu_1{}^2) + \ln(P(w_0) / P(w_1)))$$

- **δ(x)** – linear discriminant function
- **x** – input data point
- $\boldsymbol{\mu_0}$ and $\boldsymbol{\mu_1}$ – means of the two classes
- $\boldsymbol{\sigma^2}$ – common within-class variance
- **P(**$\boldsymbol{\omega_0}$**)** and **P(**$\boldsymbol{\omega_1}$**)** – prior probabilities of the two classes

## Tried other methods:

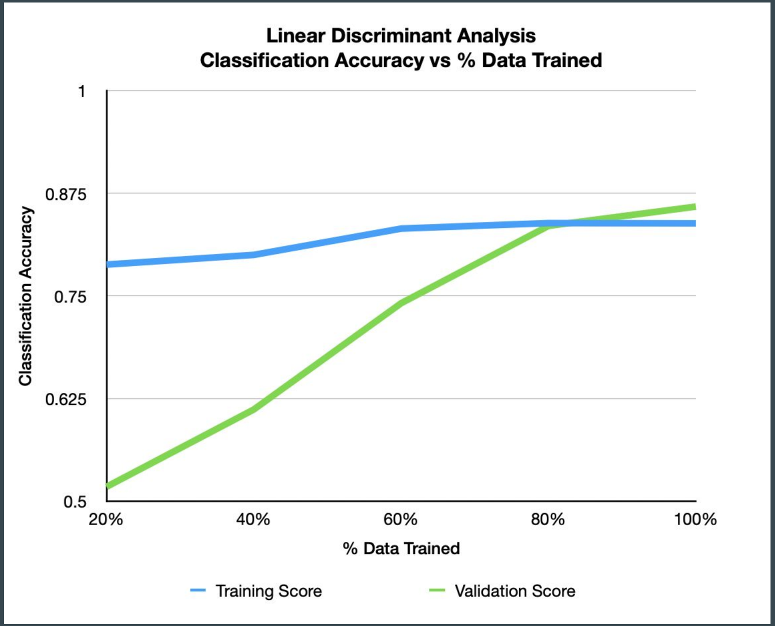
- k Nearest Neighbour
- Logistic Regression



# LDA Result

Best Result:

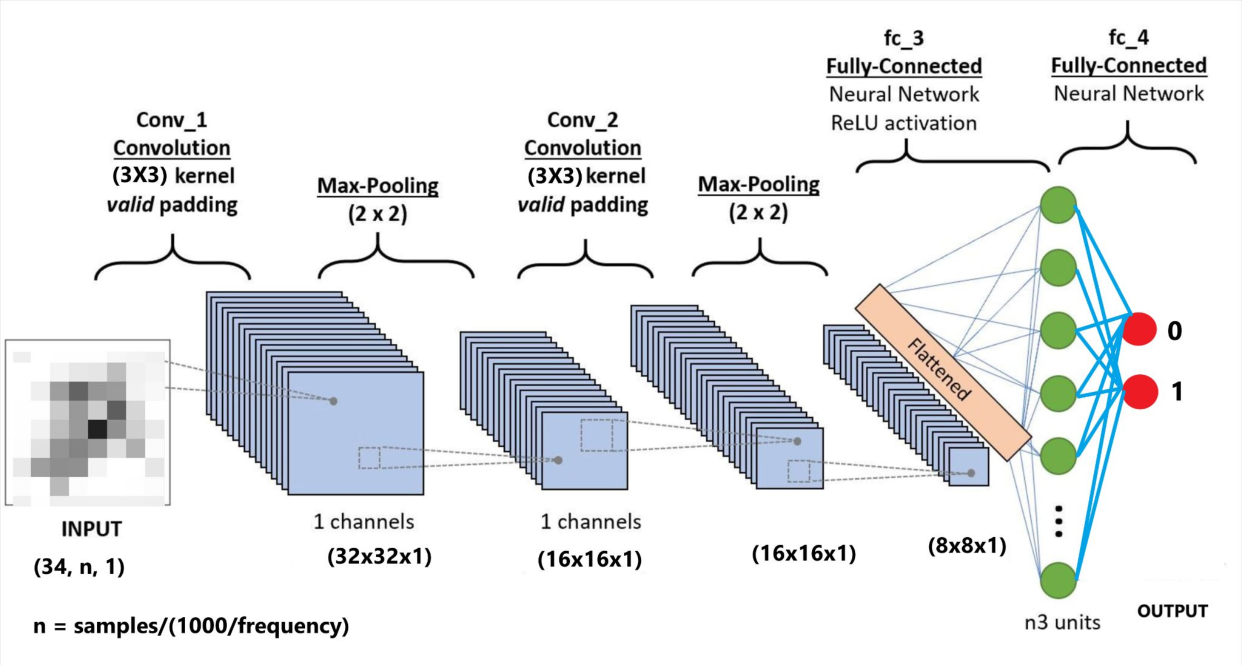
- 1620 ms event duration, resample frequency 180Hz
- Classification Accuracy
  - 83.2% training data
  - 74.1% validation data
  - 51.2% testing data



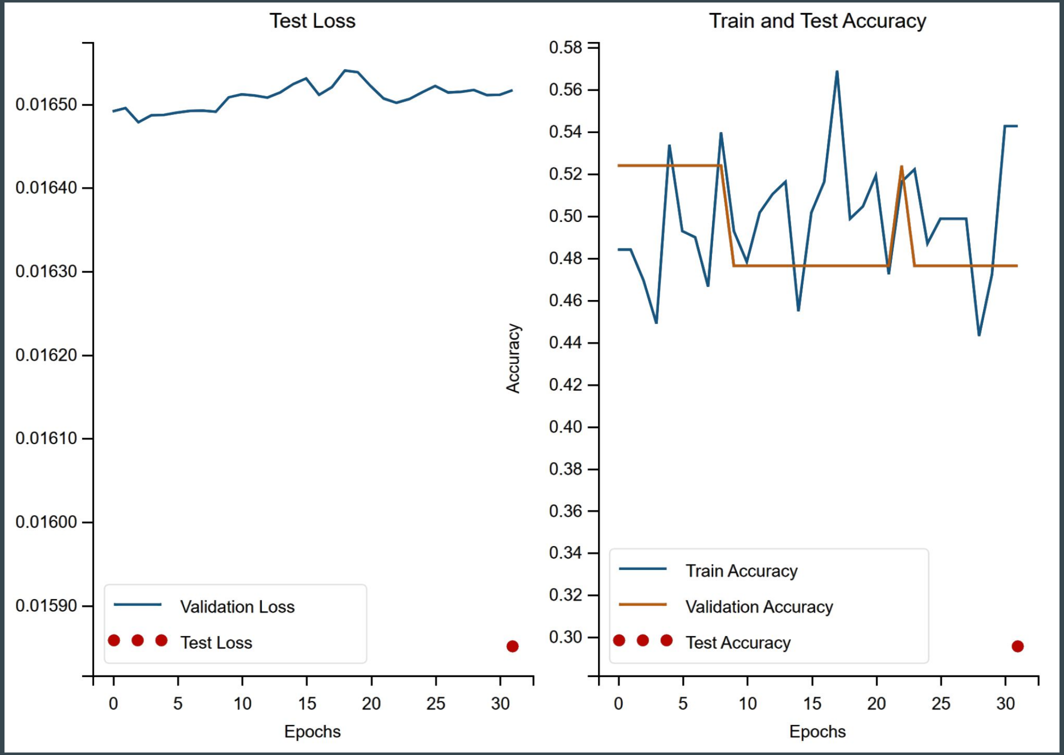

| Training<br>83.2% accuracy | Validation<br>74.1% accuracy | Testing<br>51.2% accuracy |
|---|---|---|

# Analysis - Convolutional Neural Network (CNN)

Using both TensorFlow and PyTorch Library. Since our data's shape is image-alike, the layers we'll use in CNN is convolutional and fully connected layers.



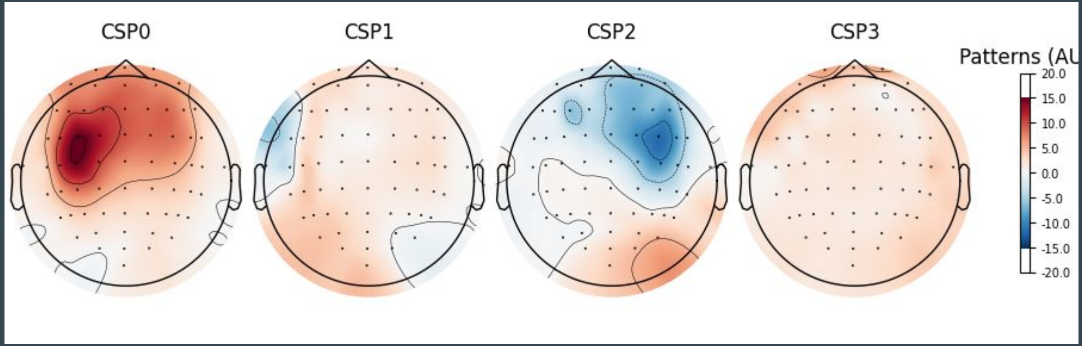
# CNN result



Test Loss
0.01650
0.01640
0.01630
0.01620
0.01610
0.01600
0.01590
Validation Loss
Test Loss
0 5 10 15 20 25 30
Epochs
Train and Test Accuracy
Accuracy
0.58
0.56
0.54
0.52
0.50
0.48
0.46
0.44
0.42
0.40
0.38
0.36
0.34
0.32
0.30
Train Accuracy
Validation Accuracy
Test Accuracy
0 5 10 15 20 25 30
Epochs

# Why CNN doesn't work

- After first convolution, we lose channels.
- Convolutional network learns how alike are the neighboring pixels and generating feature representations. Yet, there's likely little relationship between neighboring channels, thus, mixing the channels from the beginning is unsuccessful.
- The channels from 1-31 might not be anatomically similar (unlike images of body parts)

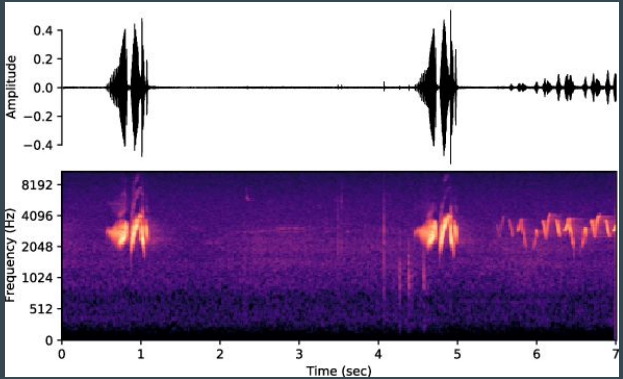


# Challenges

- Machine Learning
- EEG machine learning is niche, few examples
- MNE robust, assumes background knowledge

# Further Analysis

- Work in 1d - flatten data and use 3x1 kernels
- Principal component analysis - detect channels with the most predictive power.
- Trying other models: General Linear Model (GLM)
- Frequency decomposition.



Thank you!
Any Questions?