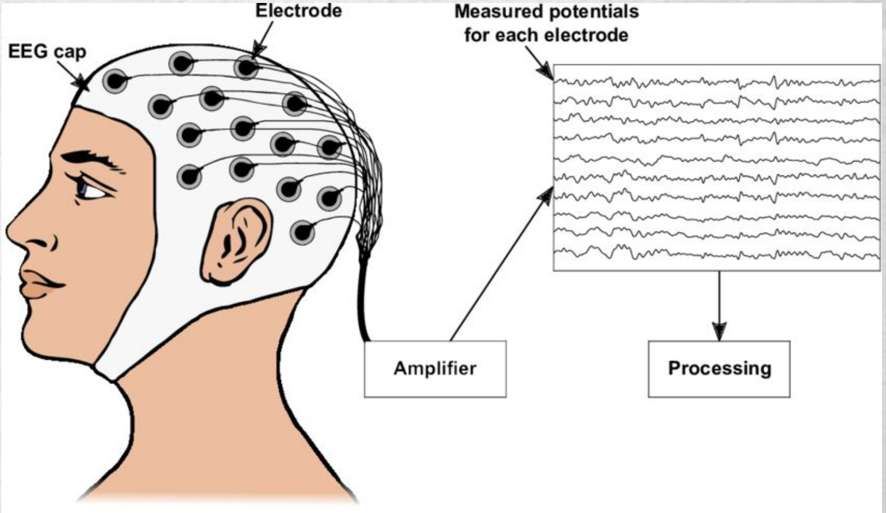
EEG classification of ADHD patients with Mutual Information and Graph Neural Network

Yongjun Lee, Harun Pirim

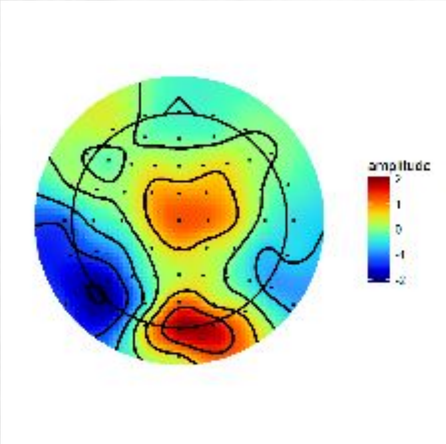
Background

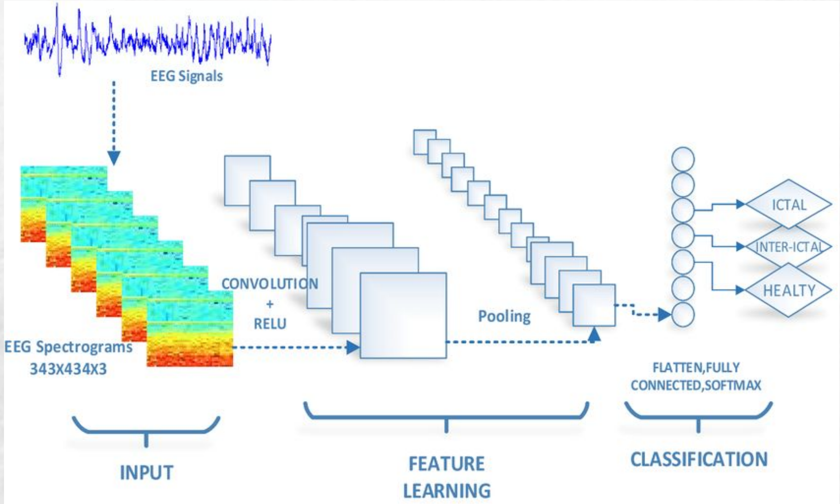
- What is EEG?
  - Electroencephalogram
  - Test that records brain activity with metal electrodes
  - Use Cases
    - BMI
    - Robot prosthetics
    - Mental illness research

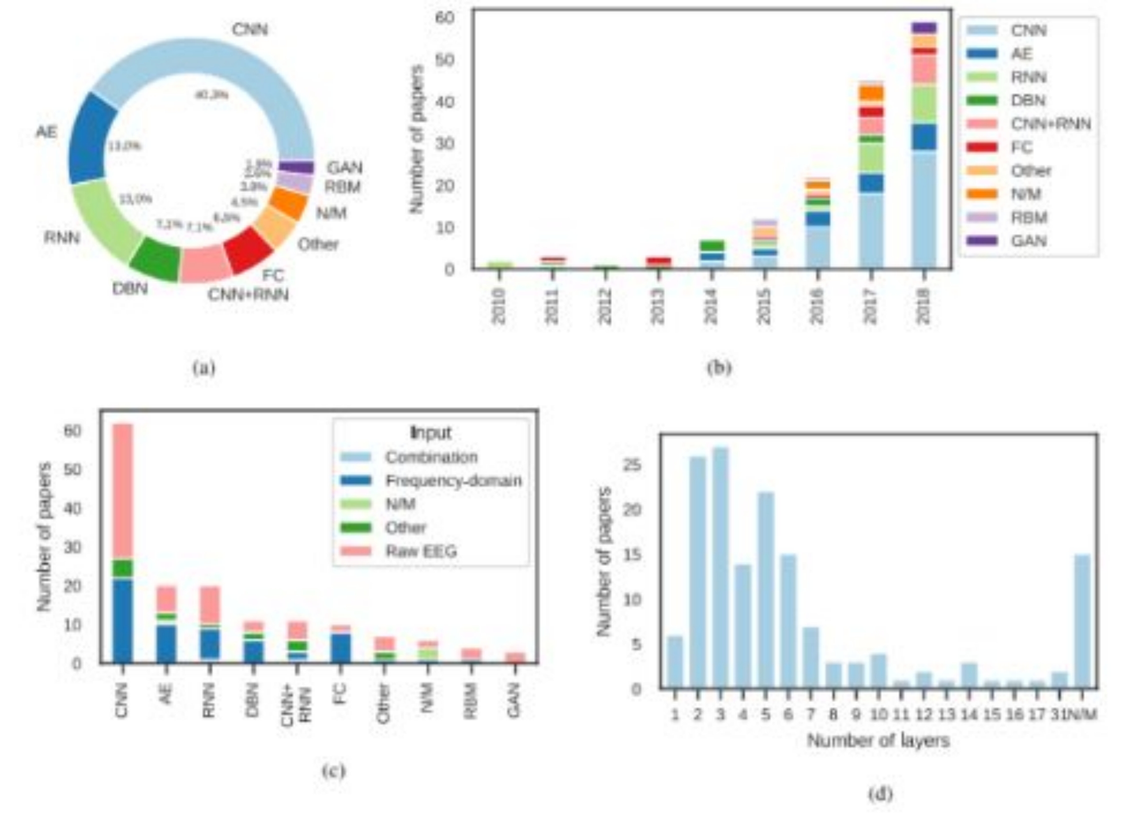


Background

- What is EEG?
- Deep Learning in EEG.
  - CNN is most widely used for EEG research







Background

- What is EEG?
- Deep Learning in EEG.
  - CNN is most widely used for EEG research
  - “Black Box” problem.

“the describing of raw EEG data without a theory driven study and standardized protocol, is problematic. It is possible that this approach is inhibiting the development of useful information regarding this potentially valuable method to aid diagnosis.”

- Adamou1 et al.

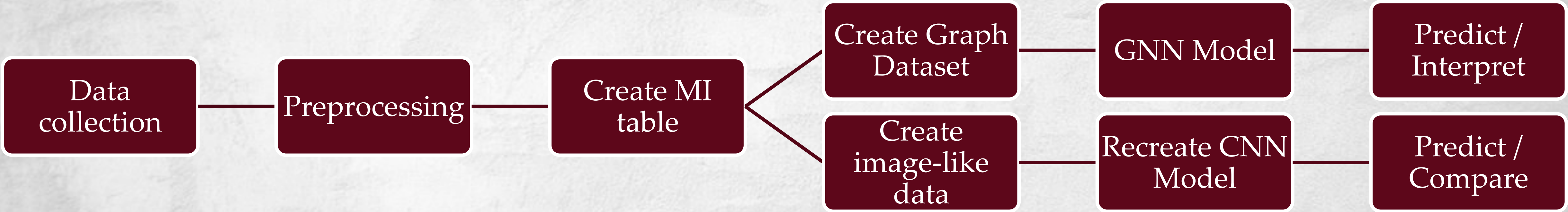
Background

- What is EEG?
- Deep Learning in EEG.
- Graph Neural Network
  - GNN suits EEG dataset more naturally.
  - Better performance, interpretability.

Related Work

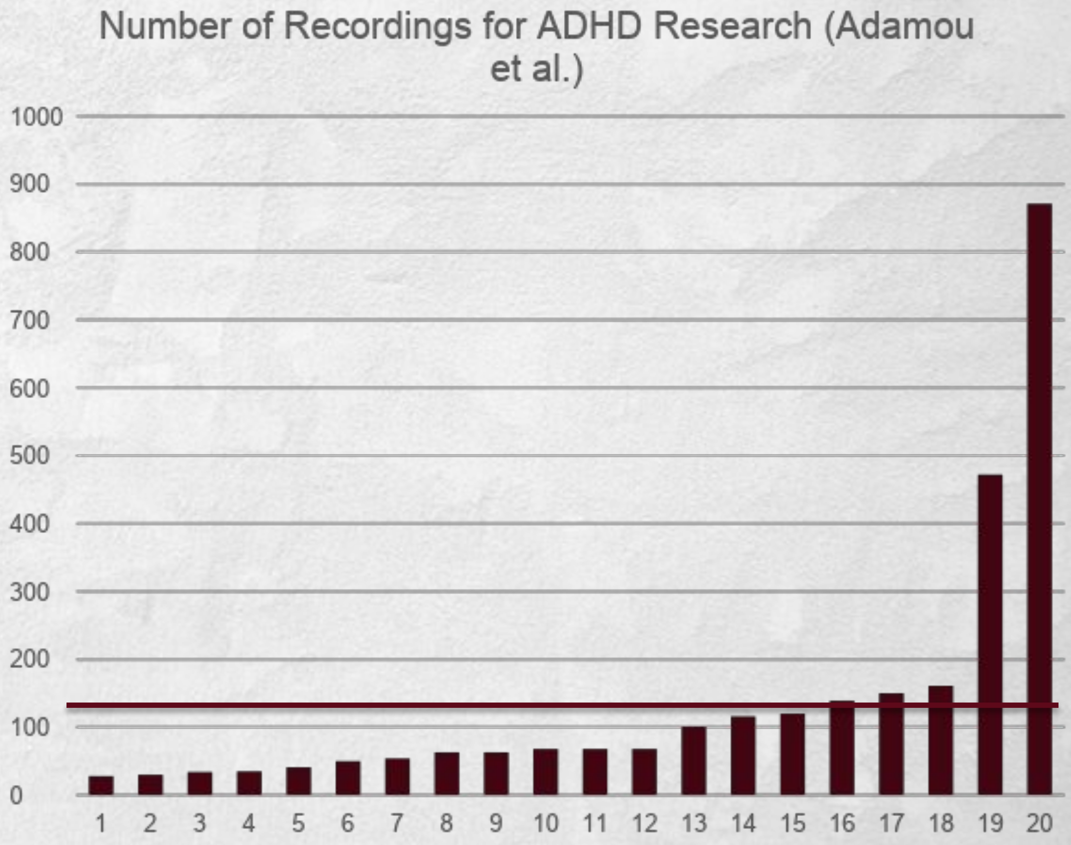
- Proposed model from "A deep learning framework for identifying children with ADHD using an EEG-based brain network"
  - MI + CNN
- Demir et al. reviewed six GNN models and benchmarked their performance for EEG classification task.
  - GraphSAGE, Graph Isomorphism Network (GIN), SortPool, EdgePool, SagPool, Set2Set

Method Overview



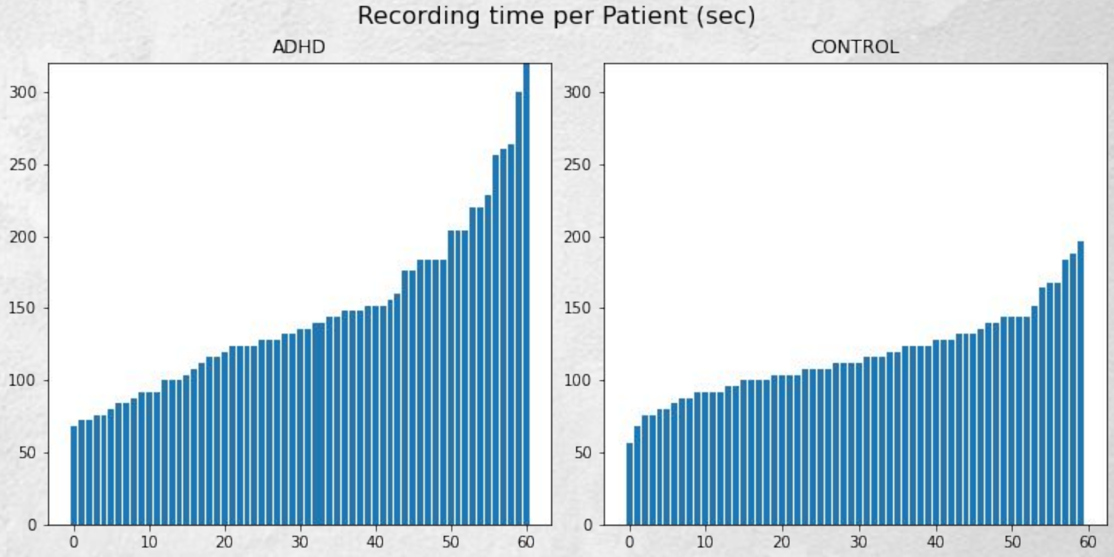
Data Collection

- IEEE Dataport
- 61 ADHD recordings, 60 Control group recordings
  - Boys and girls, ages 7-12
- 128 Hz, 54~380 seconds per recording
- 19 channels



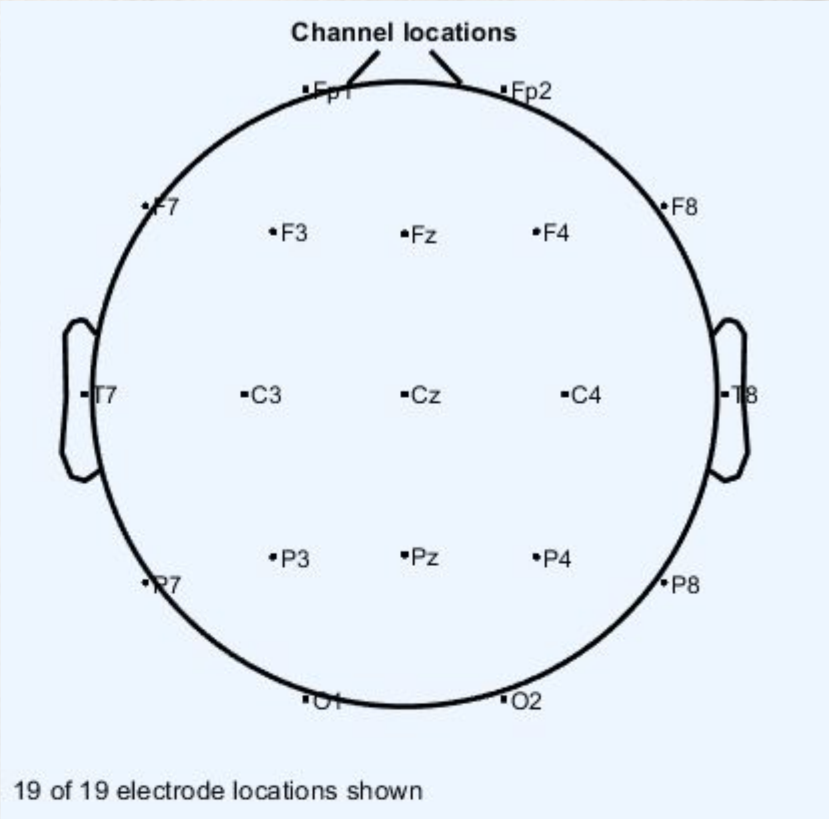
Data Collection

- IEEE Dataport
- 61 ADHD recordings, 60 Control group recordings
  - Boys and girls, ages 7-12
- 128 Hz, 54~380 seconds per recording
- 19 channels



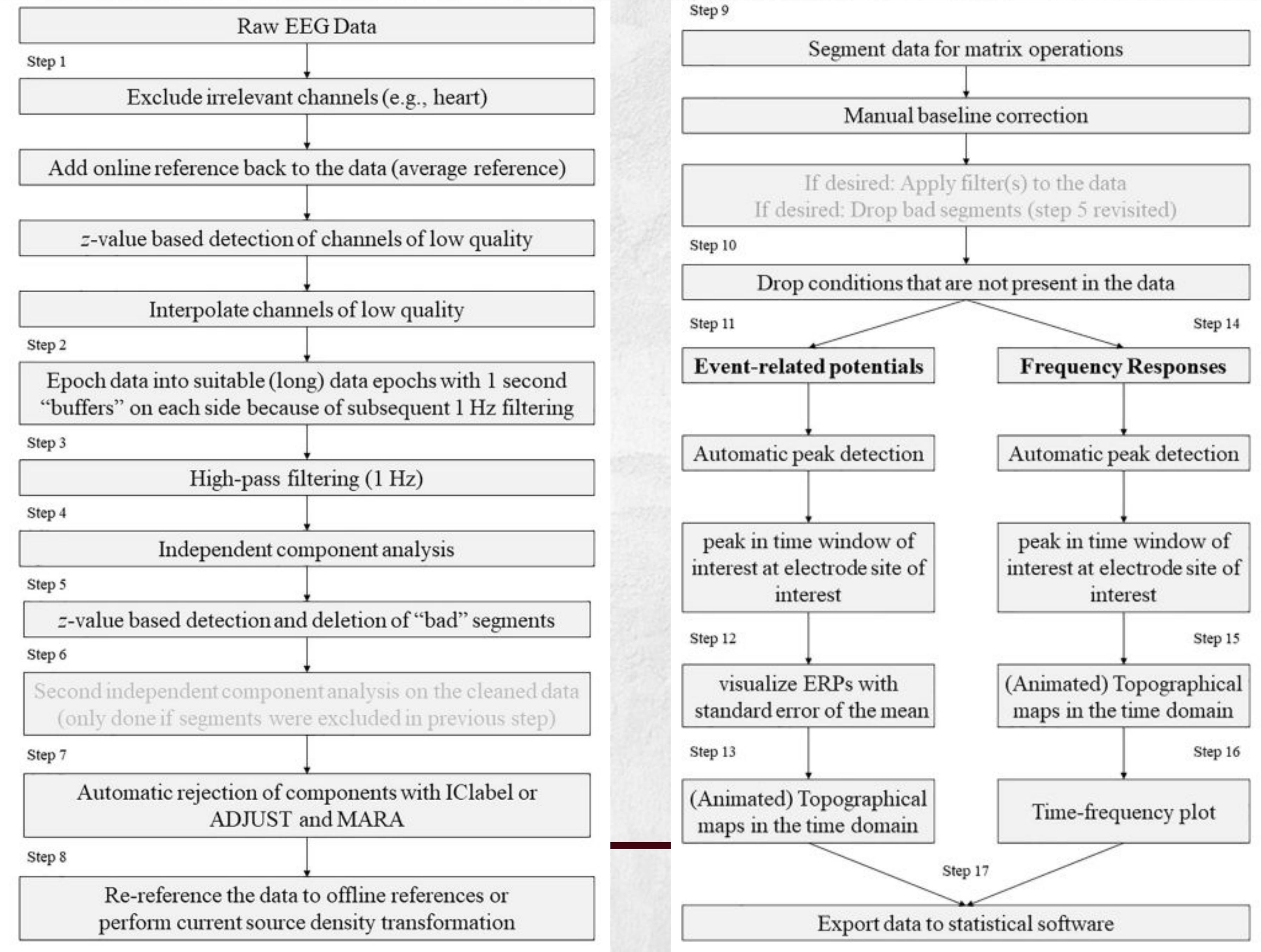
Data Collection

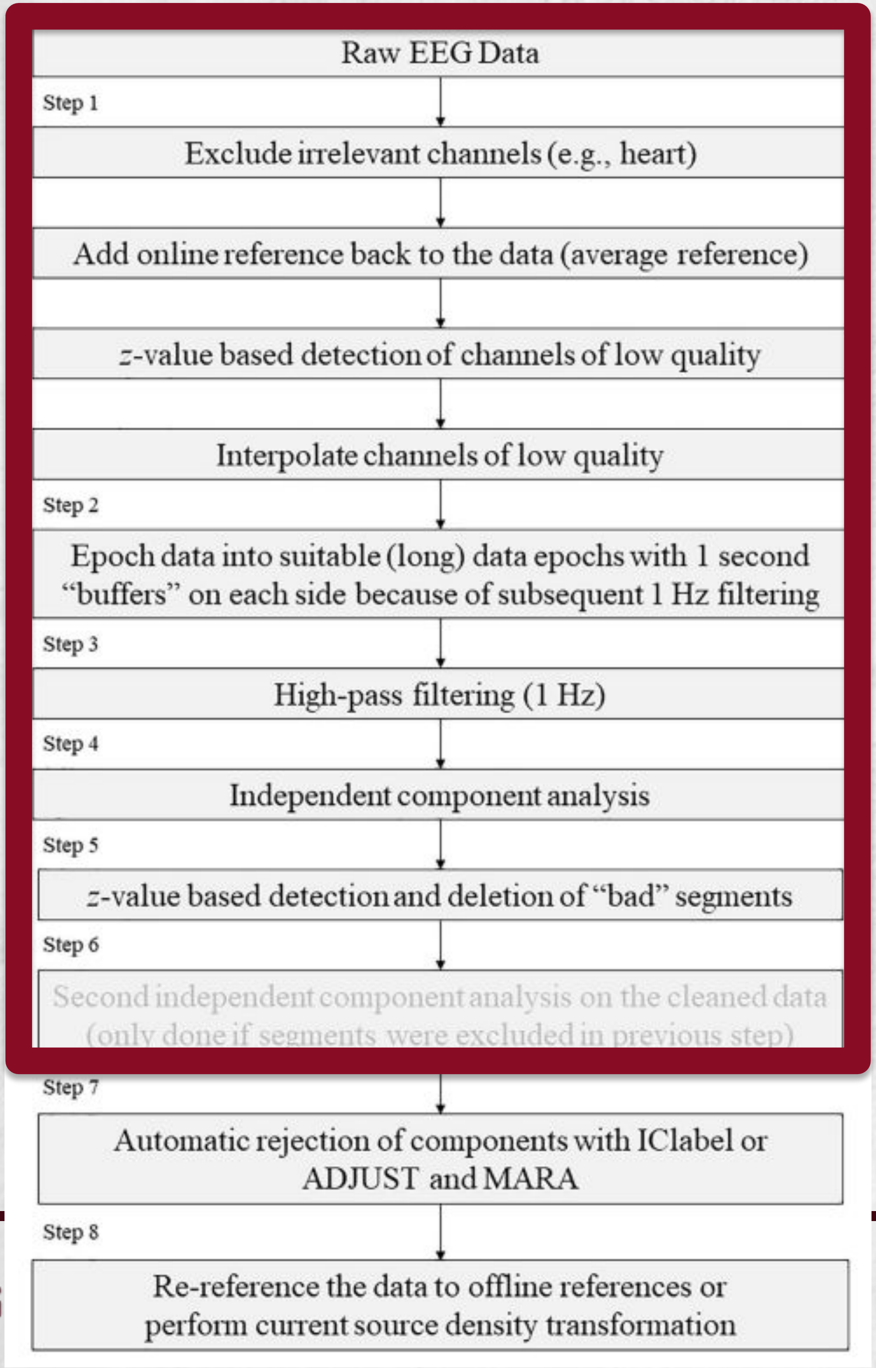
- IEEE Dataport
- 61 ADHD recordings, 60 Control group recordings
  - Boys and girls, ages 7-12
- 128 Hz, 54~380 seconds per recording
- 19 channels



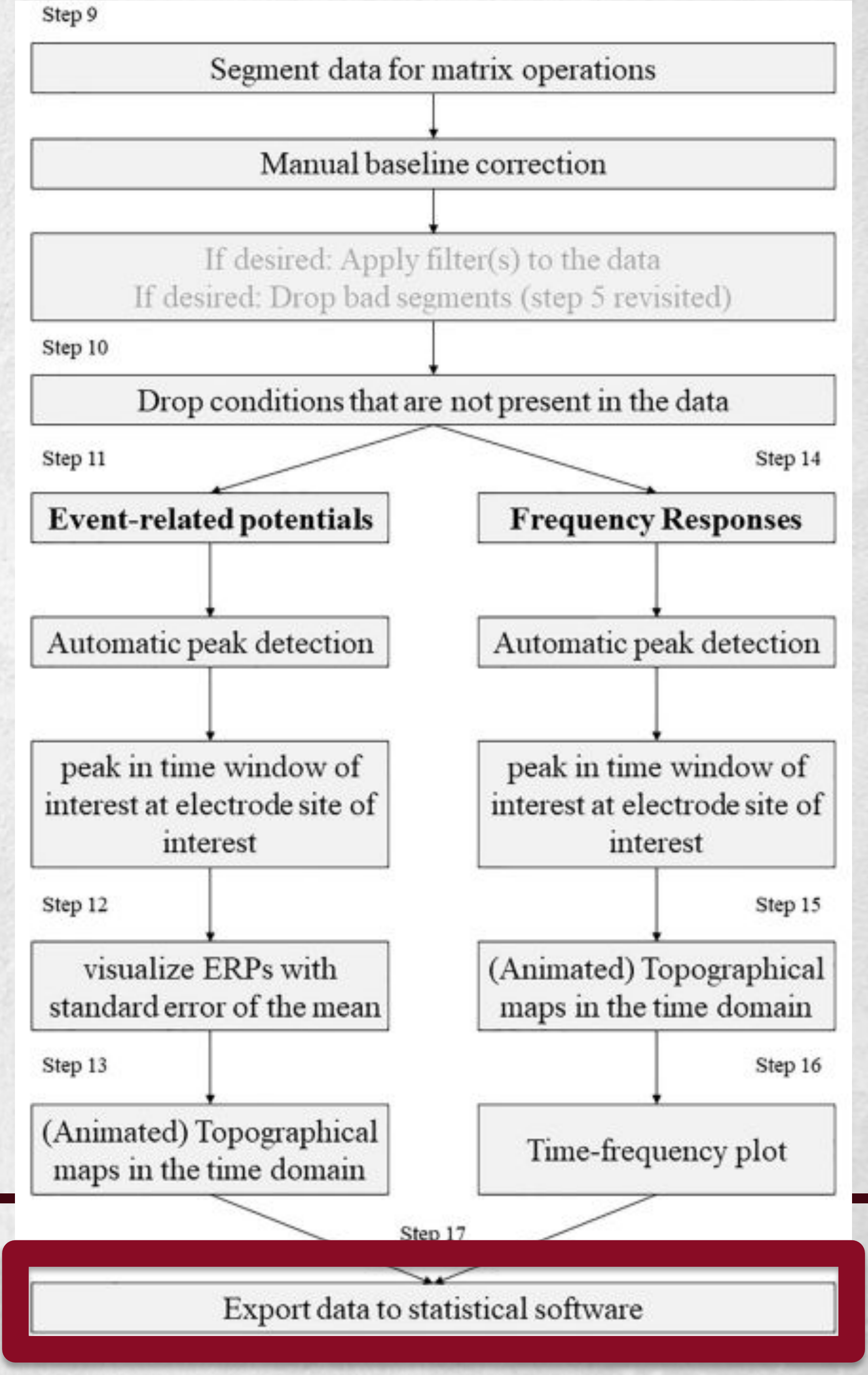
Preprocessing

- EEGLAB (Delorme & Makeig, 2004) is a widely used EEG processing tool
  - Built on MATLAB
- EPOS is an EEG preprocessing pipeline (Rodrigues et al.)
  - Built using EEGLAB

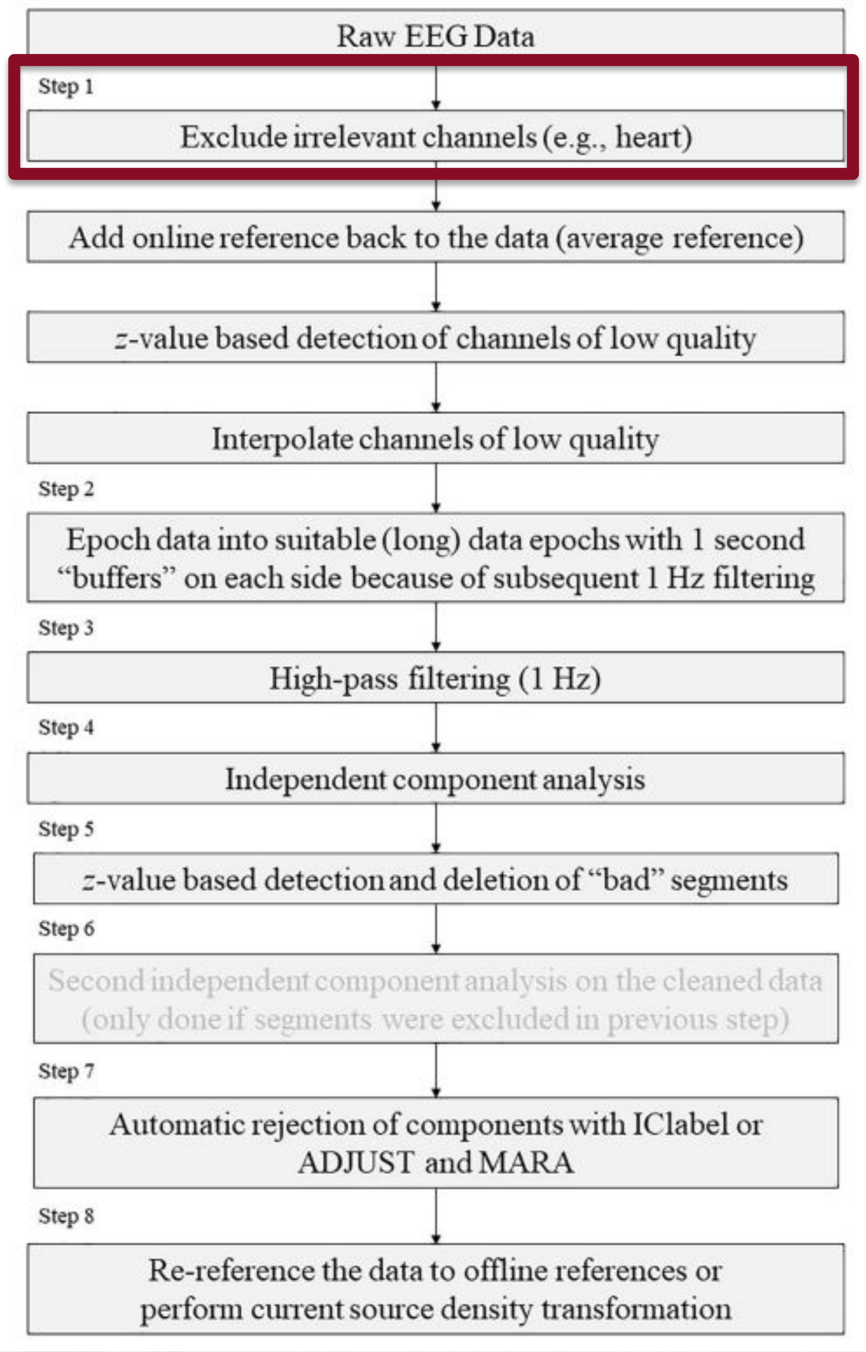

Raw EEG Data
Step 1
Exclude irrelevant channels (e.g., heart)
Add online reference back to the data (average reference)
z-value based detection of channels of low quality
Interpolate channels of low quality
Step 2
Epoch data into suitable (long) data epochs with 1 second "buffers" on each side because of subsequent 1 Hz filtering
Step 3
High-pass filtering (1 Hz)
Step 4
Independent component analysis
Step 5
z-value based detection and deletion of "bad" segments
Step 6
Second independent component analysis on the cleaned data (only done if segments were excluded in previous step)
Step 7
Automatic rejection of components with IClabel or ADJUST and MARA
Step 8
Re-reference the data to offline references or perform current source density transformation
Step 9
Segment data for matrix operations
Manual baseline correction
If desired: Apply filter(s) to the data
If desired: Drop bad segments (step 5 revisited)
Step 10
Drop conditions that are not present in the data
Step 11
Step 14
Event-related potentials
Frequency Responses
Automatic peak detection
Automatic peak detection
peak in time window of interest at electrode site of interest
peak in time window of interest at electrode site of interest
Step 12
Step 15
visualize ERPs with standard error of the mean
(Animated) Topographical maps in the time domain
Step 13
Step 16
(Animated) Topographical maps in the time domain
Time-frequency plot
Step 17
Export data to statistical software



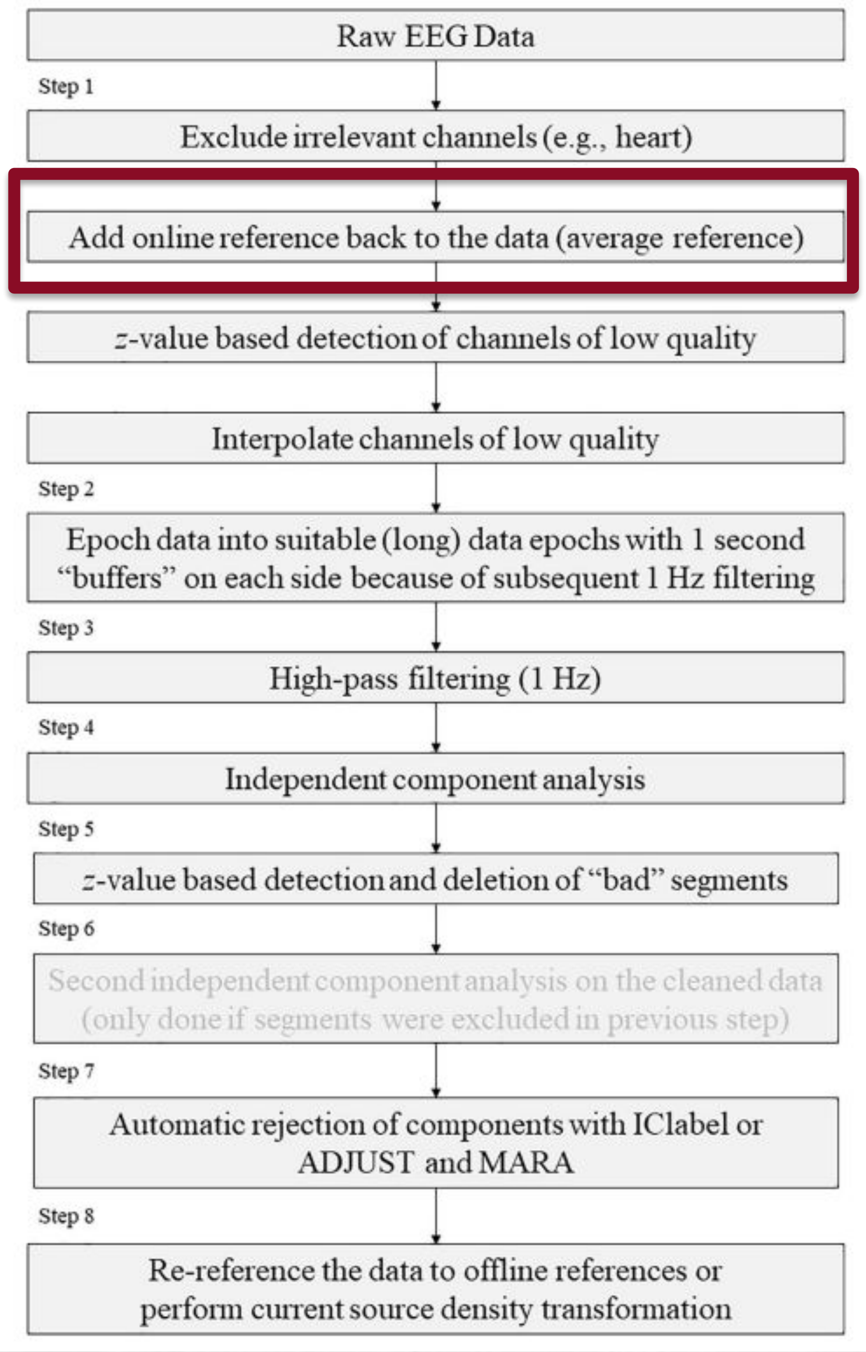
Raw EEG Data
Step 1
Exclude irrelevant channels (e.g., heart)
Add online reference back to the data (average reference)
z-value based detection of channels of low quality
Interpolate channels of low quality
Step 2
Epoch data into suitable (long) data epochs with 1 second "buffers" on each side because of subsequent 1 Hz filtering
Step 3
High-pass filtering (1 Hz)
Step 4
Independent component analysis
Step 5
z-value based detection and deletion of "bad" segments
Step 6
Second independent component analysis on the cleaned data (only done if segments were excluded in previous step)
Step 7
Automatic rejection of components with IClabel or ADJUST and MARA
Step 8
Re-reference the data to offline references or perform current source density transformation



Step 9
Segment data for matrix operations
Manual baseline correction
If desired: Apply filter(s) to the data
If desired: Drop bad segments (step 5 revisited)
Step 10
Drop conditions that are not present in the data
Step 11
Step 14
Event-related potentials
Frequency Responses
Automatic peak detection
Automatic peak detection
peak in time window of interest at electrode site of interest
peak in time window of interest at electrode site of interest
Step 12
Step 15
visualize ERPs with standard error of the mean
(Animated) Topographical maps in the time domain
Step 13
Step 16
(Animated) Topographical maps in the time domain
Time-frequency plot
Step 17
Export data to statistical software



Not necessary – No heart signal recorded

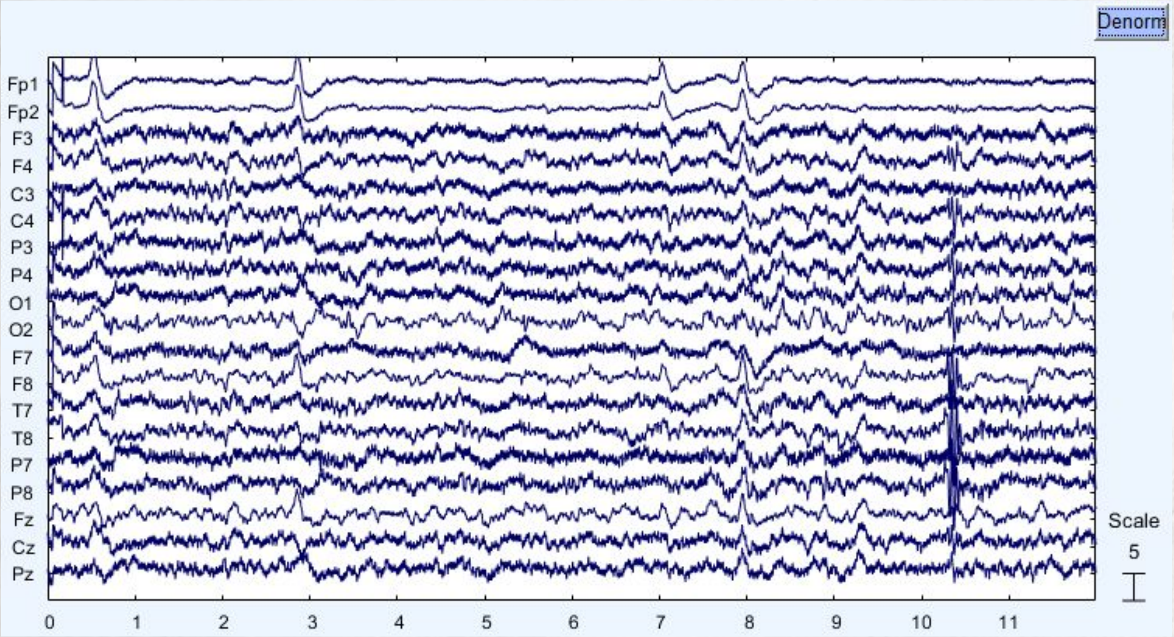


Average Reference

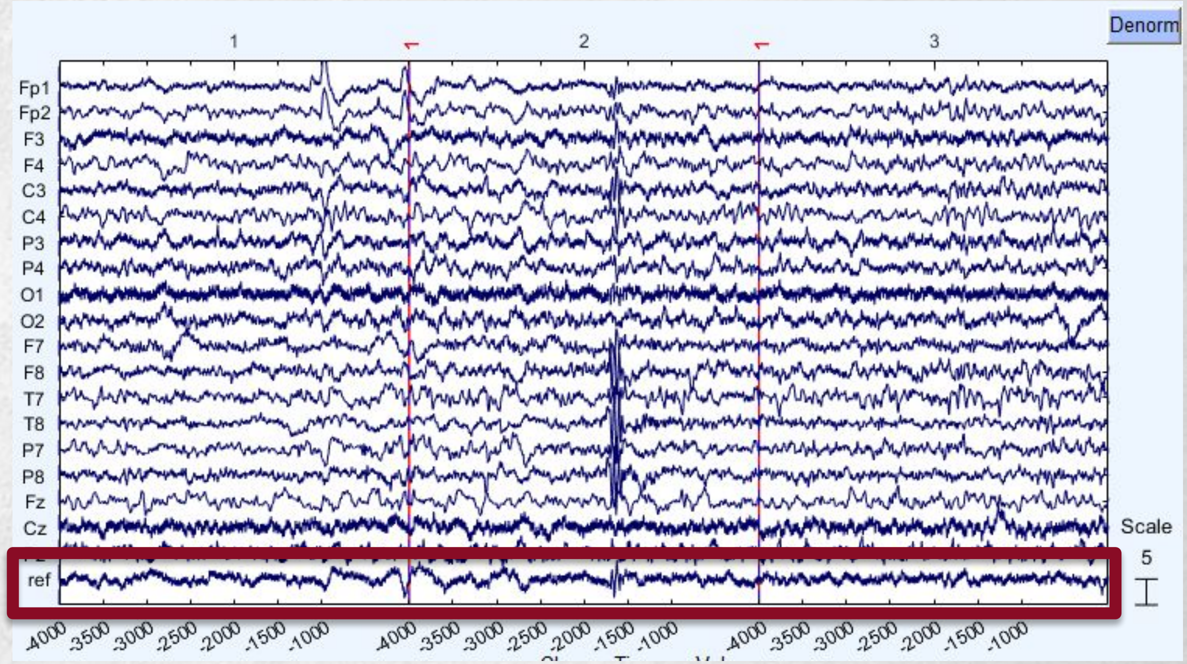
Create a new channel which contains average value of all channels

This is used for rejecting bad segments (epochs)

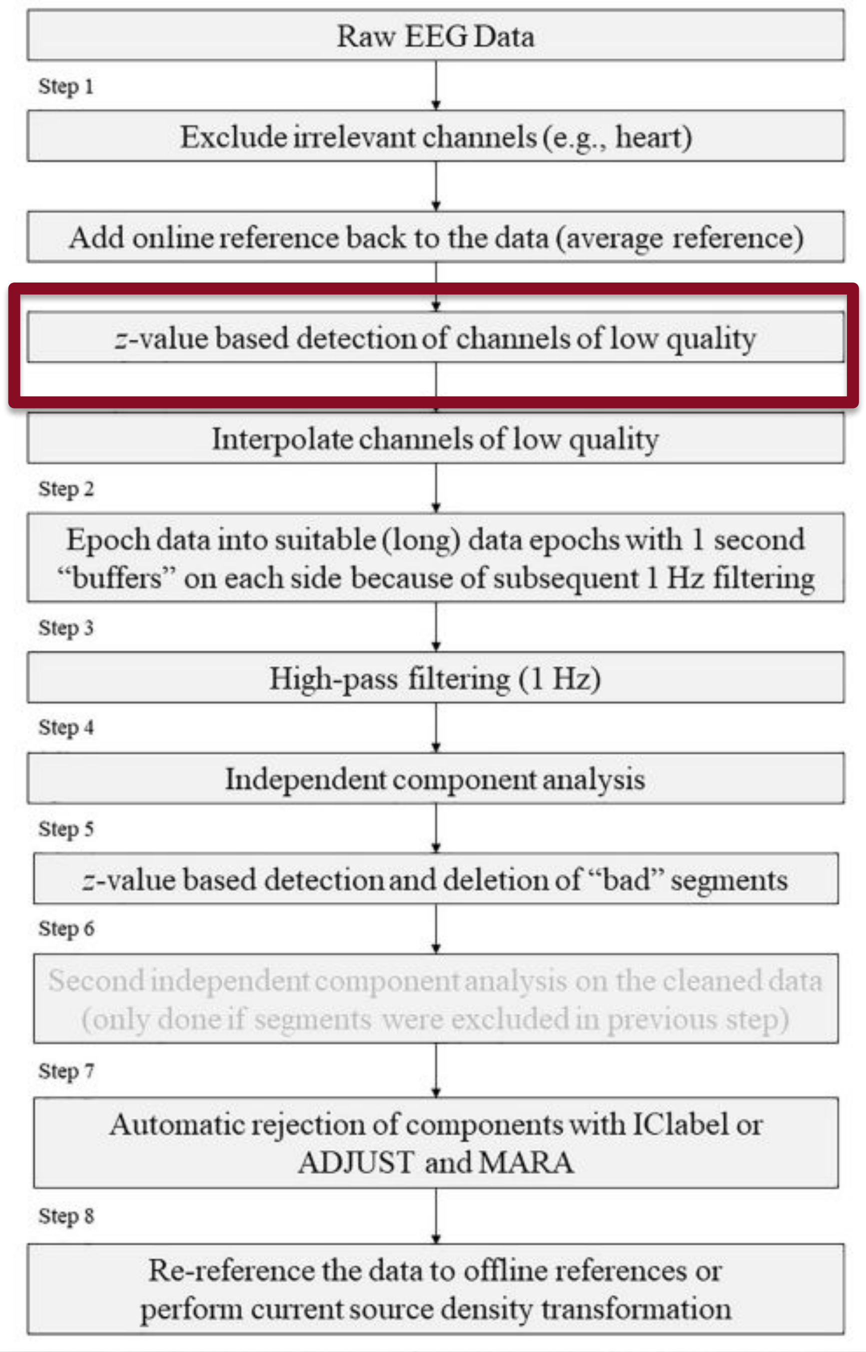
Method - Preprocessing



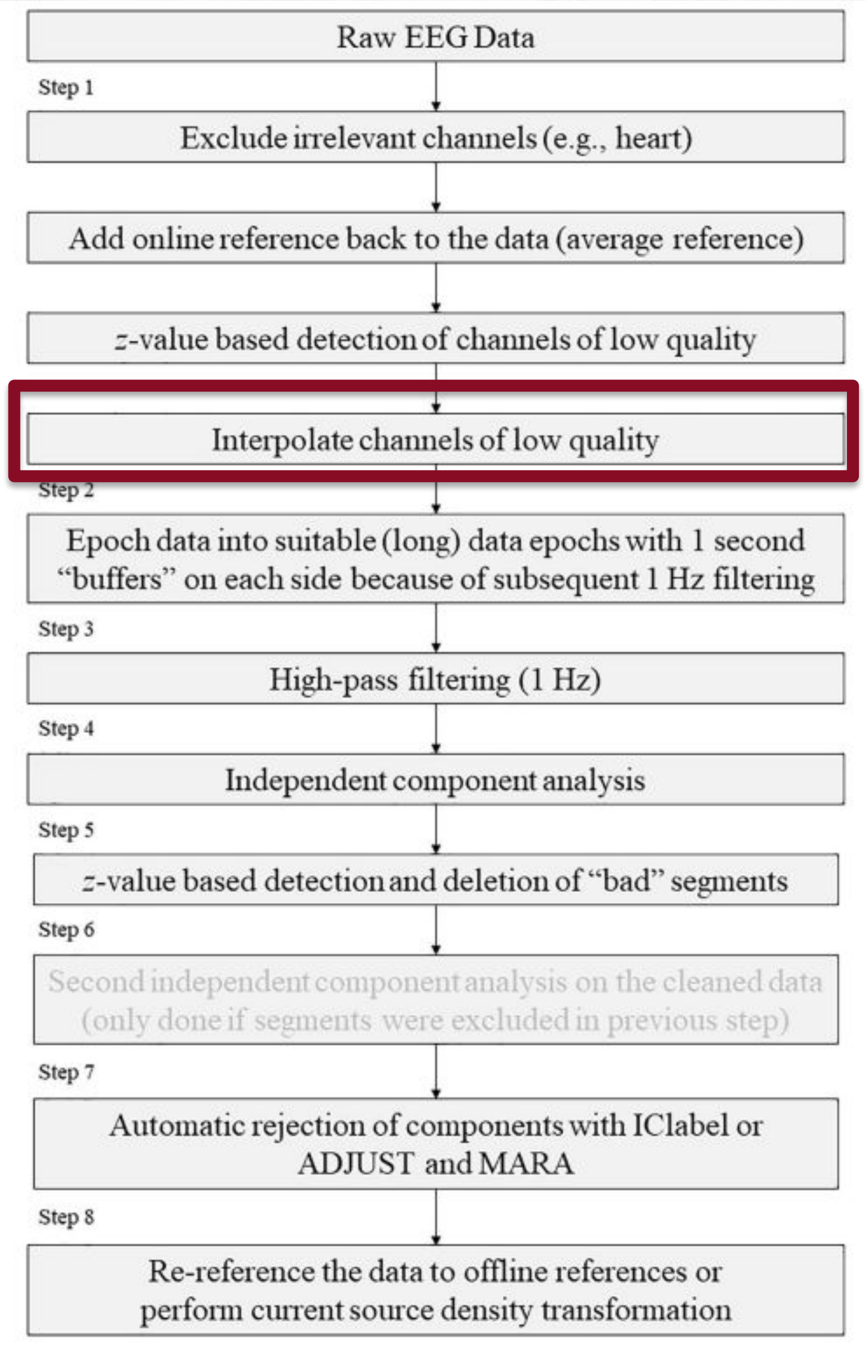
<Raw EEG signal>



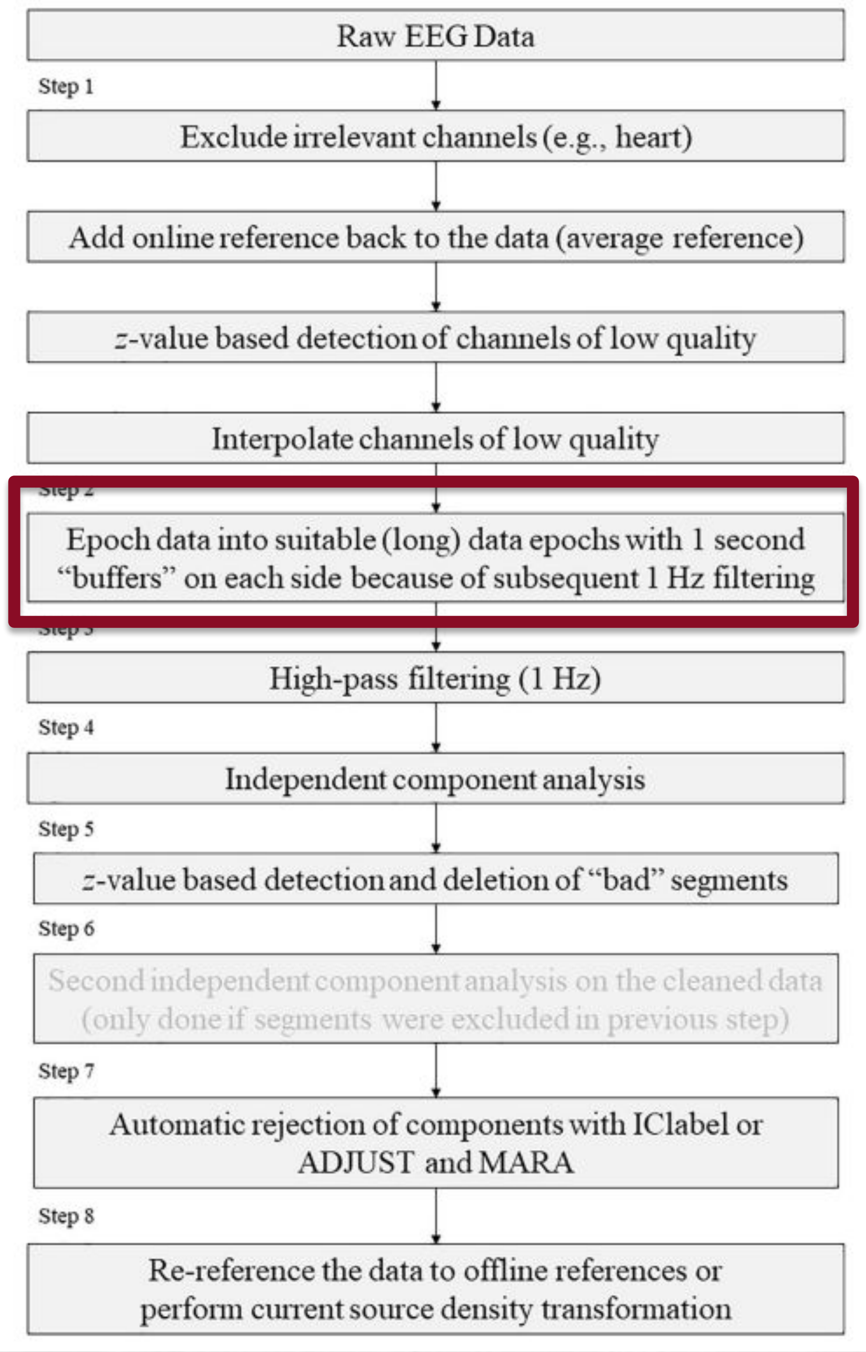
<Preprocessed EEG signal>



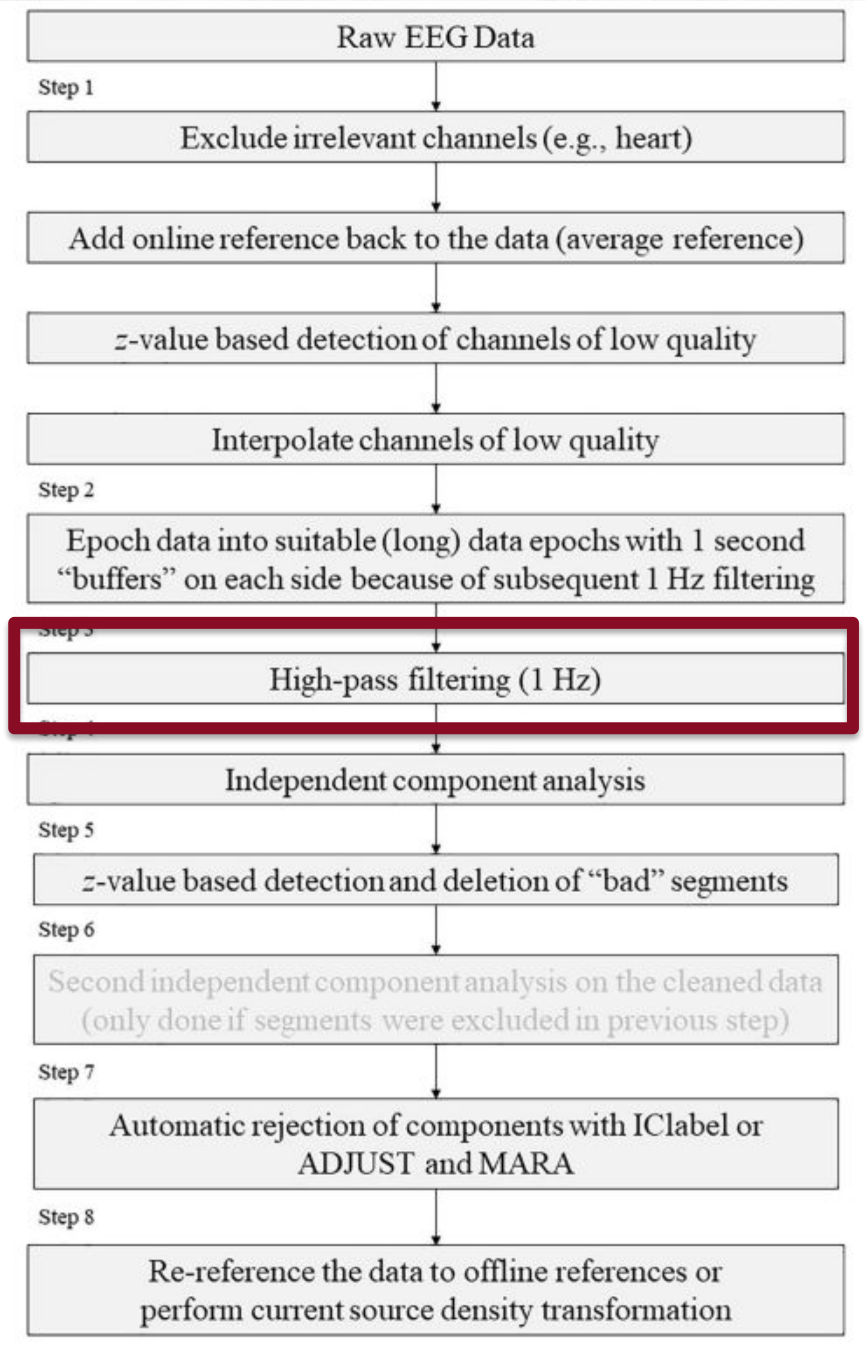
Z-test rejection
Use statistical inference with calculated average to reject a channel with bad quality.
(probability > 3.29)



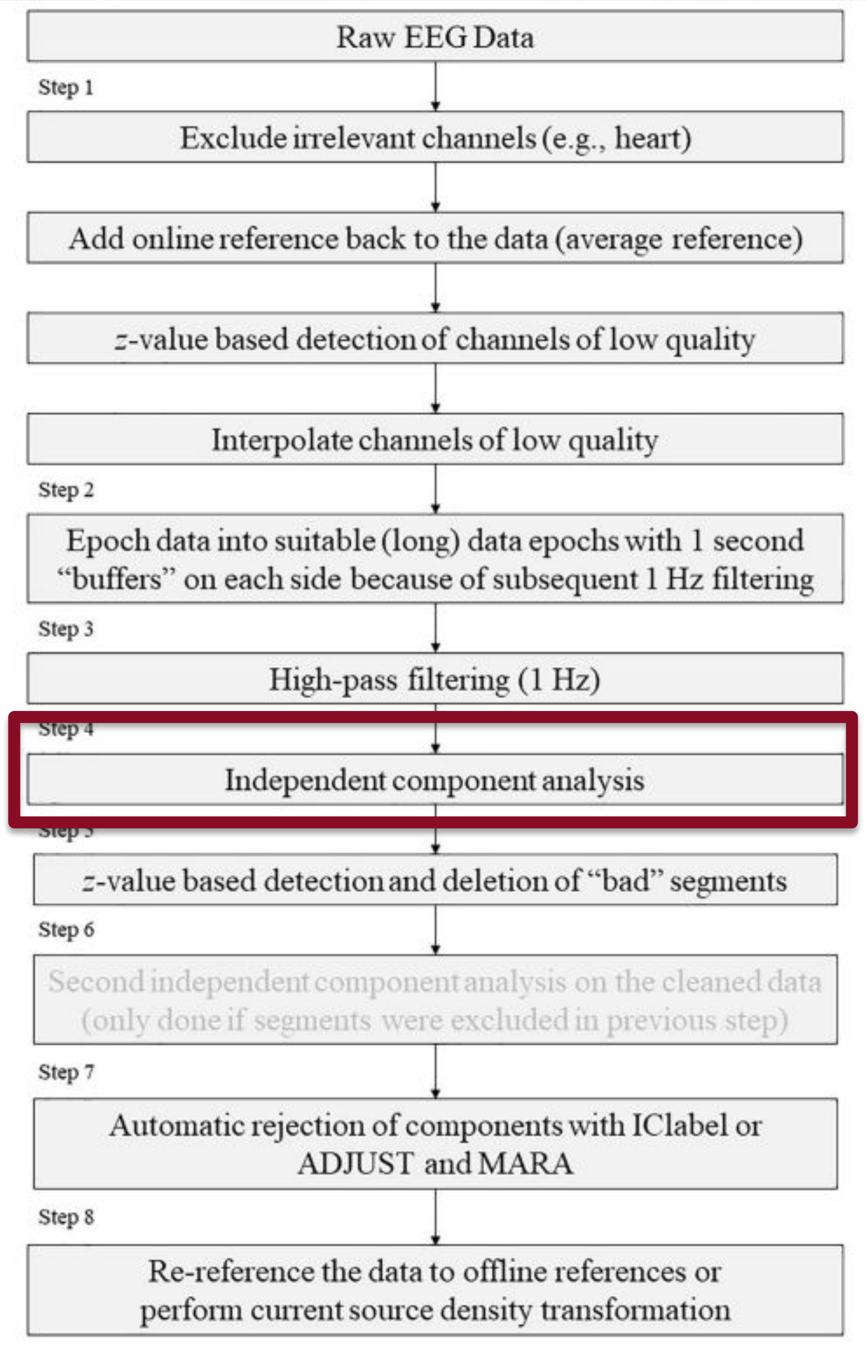
Interpolate rejected channels
Use channels around the bad channel to interpolate the values. (Spherical Interpolation)



Epoch 4 second window. 0 second overlap

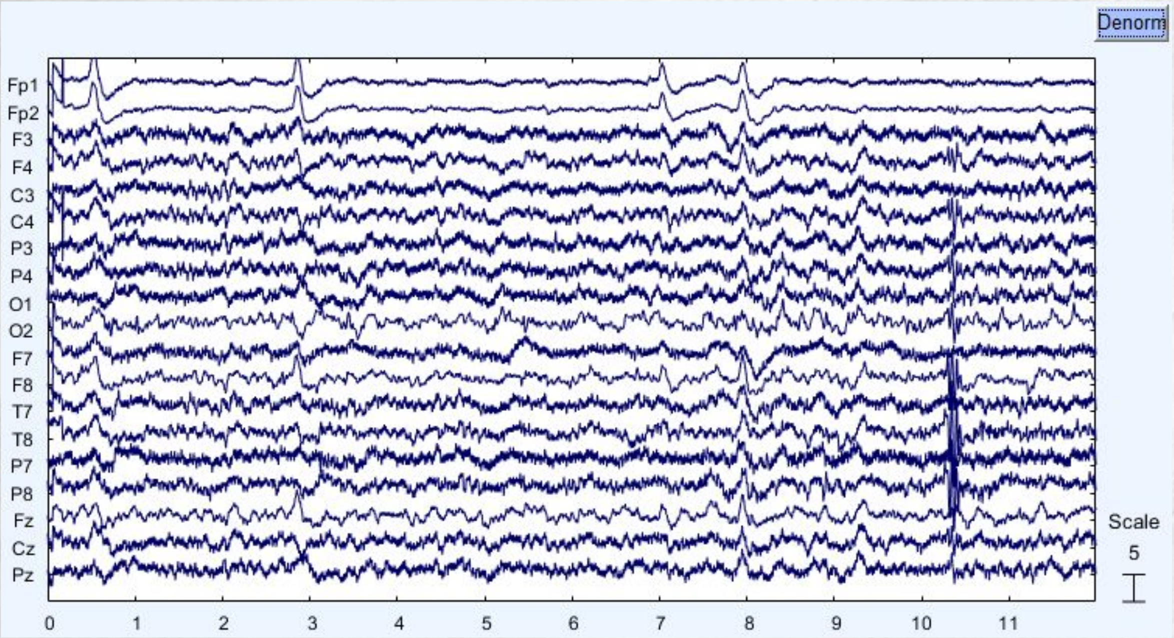


High pass filtering
All frequencies below the target frequency will be dampened
Used for stable ICA results

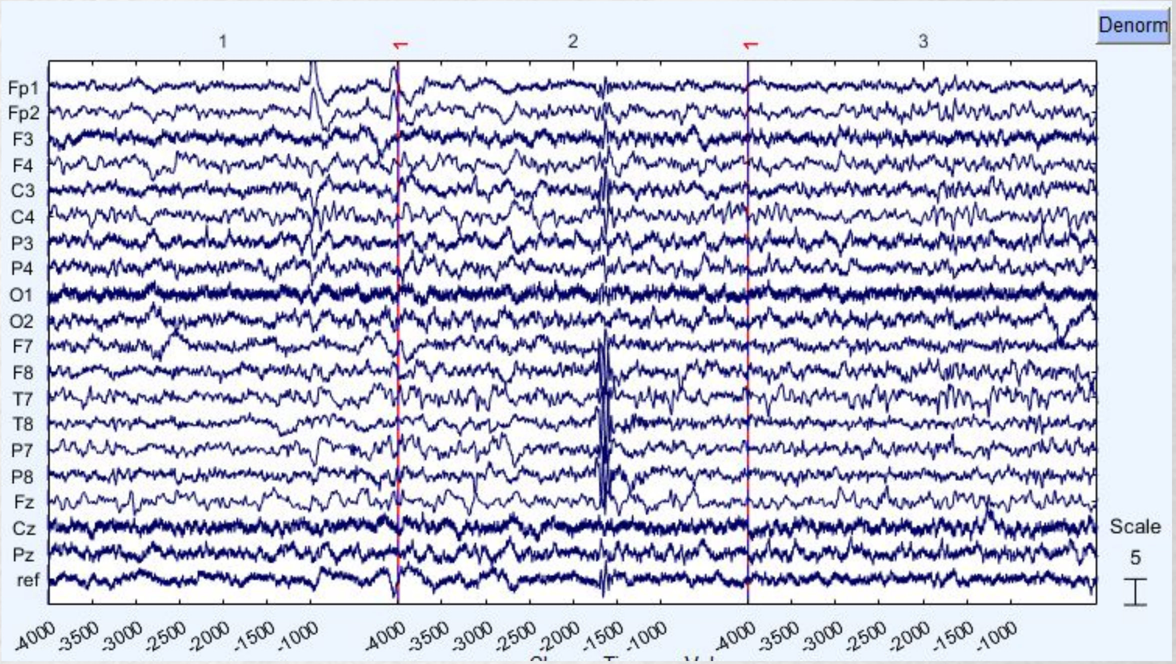


Independent Component Analysis
"ICA separates the actual electronic brain signal from non-brain artifacts such as eye movements or muscle activity."

Method - Preprocessing

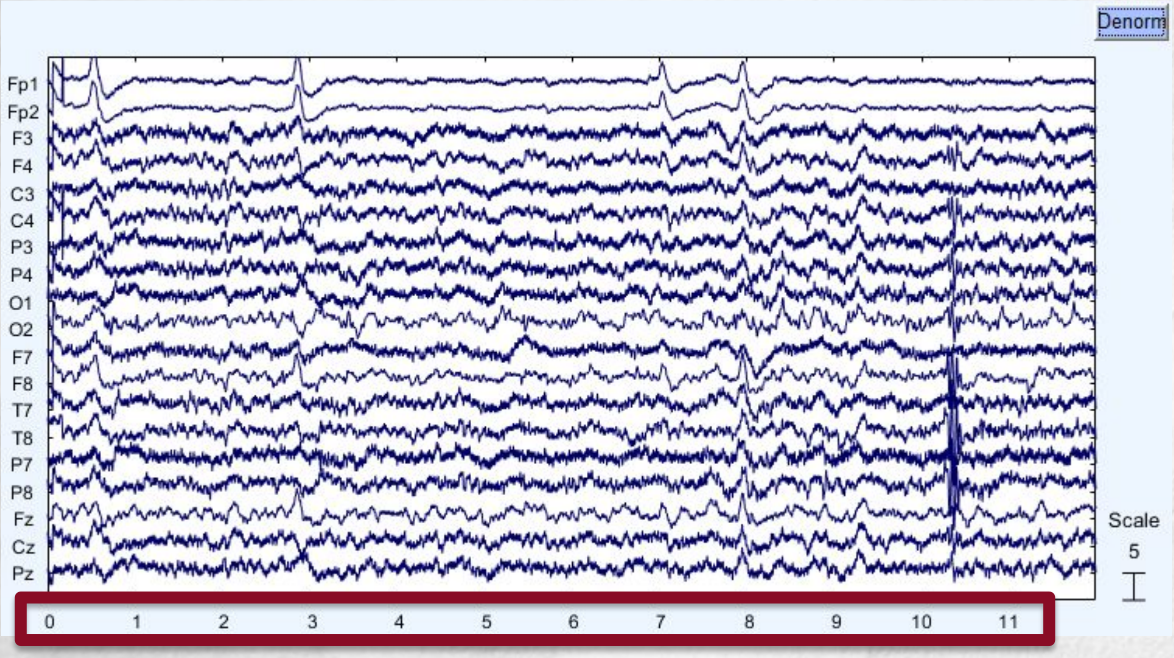


<Raw EEG signal>

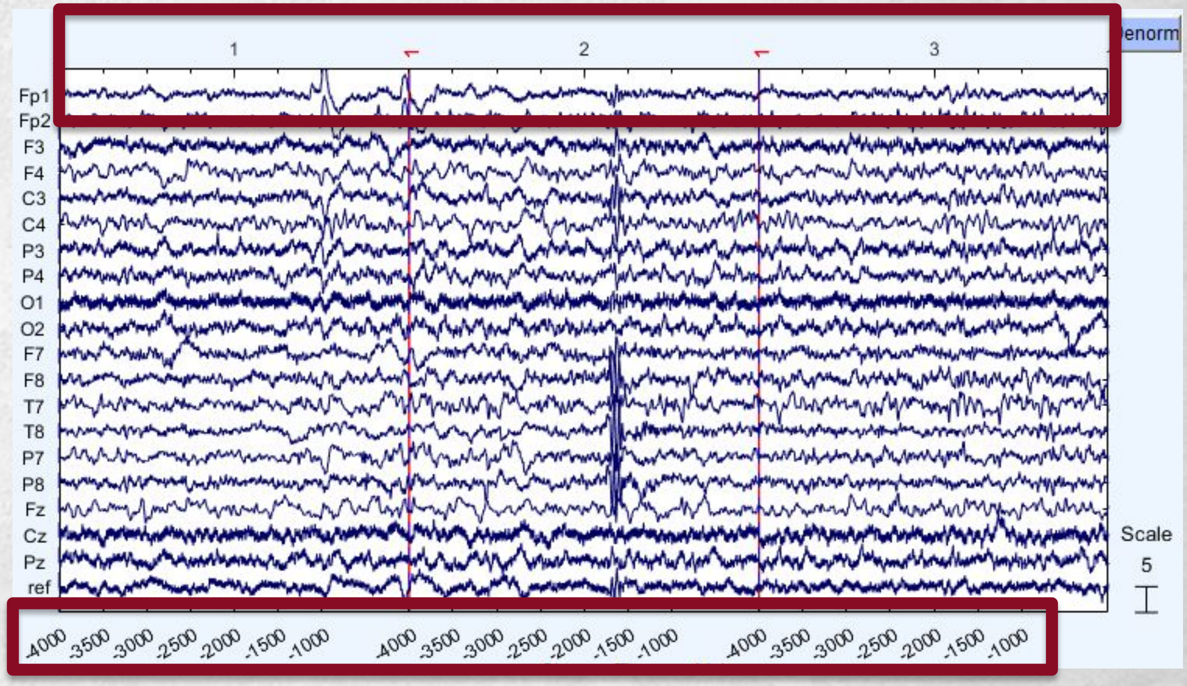


<Preprocessed EEG signal>

Method - Preprocessing

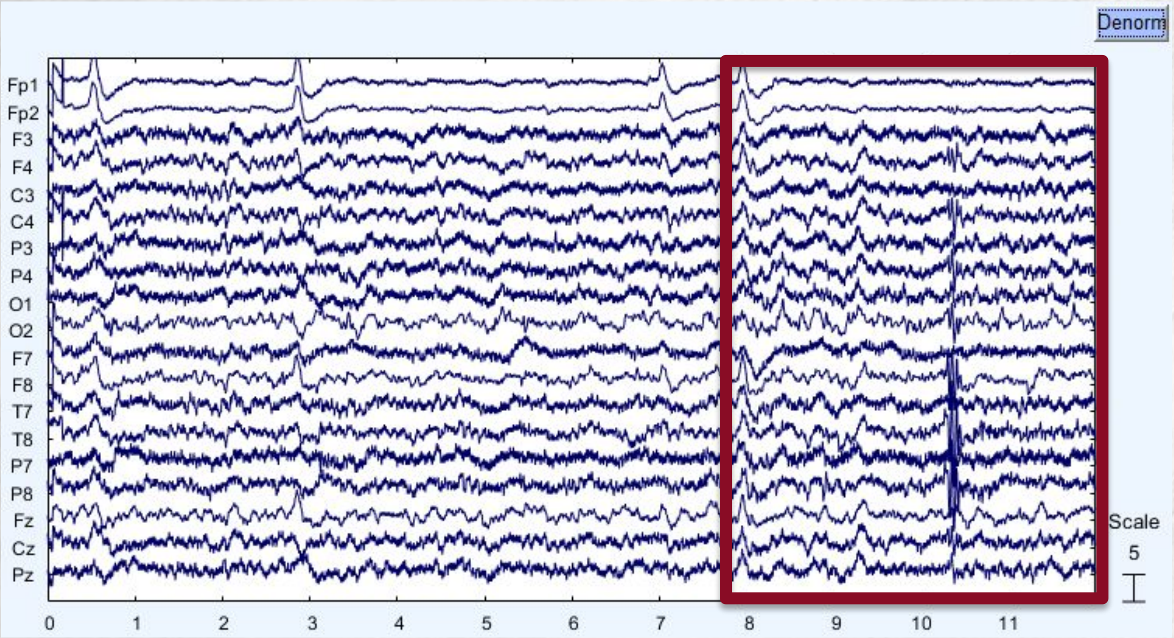


<Raw EEG signal>

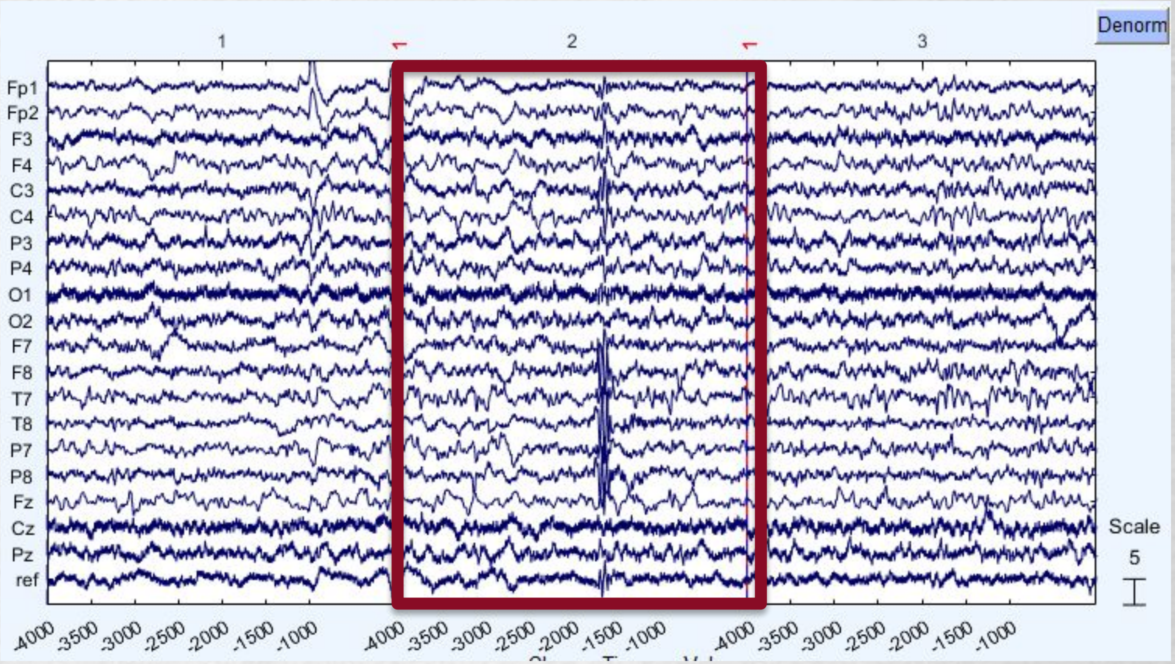


<Preprocessed EEG signal>

Method - Preprocessing



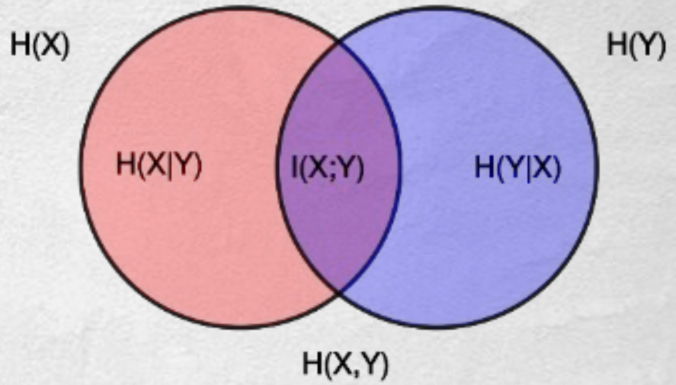
<Raw EEG signal>



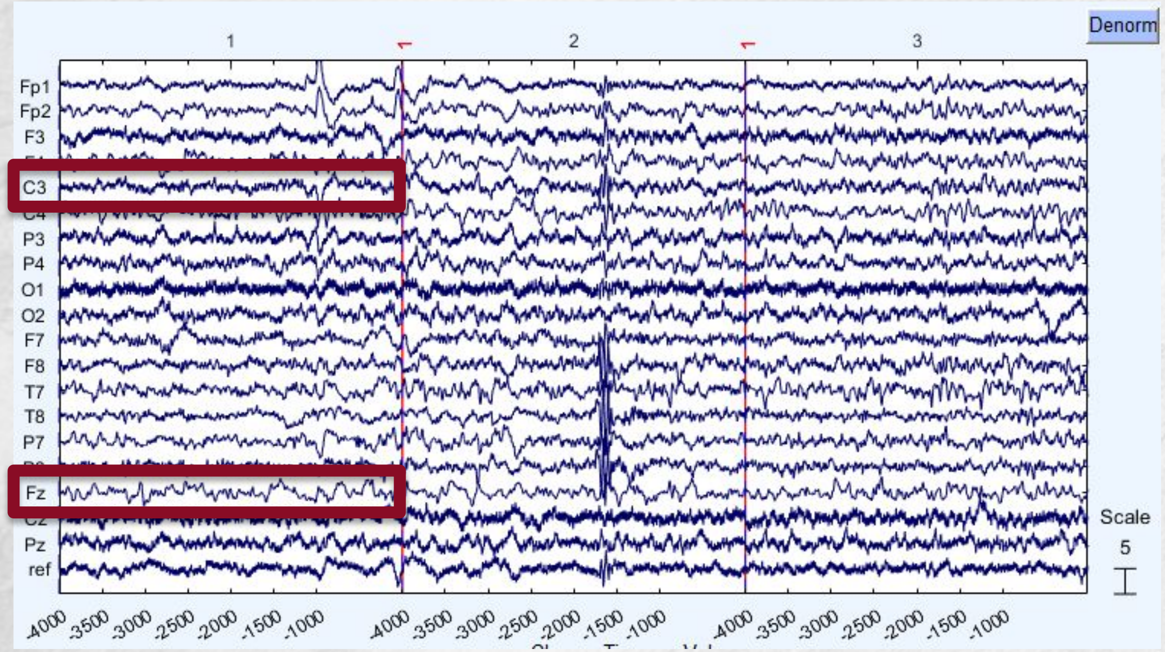
<Preprocessed EEG signal>

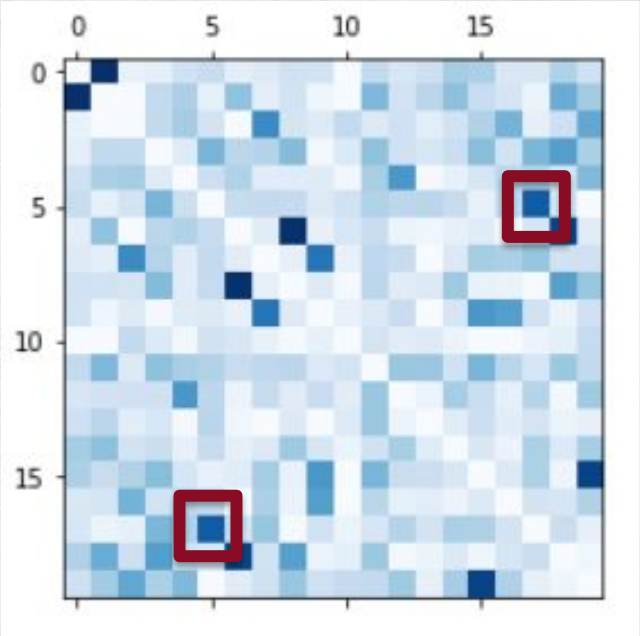
Create Mutual Information Table

- $I(X;Y) = H(X) - H(X|Y)$ (H: marginal entropy)
$$= H(Y) - H(Y|X)$$
$$= H(X,Y) - H(X|Y) - H(Y|X)$$
$$= \iint_{x,y} P_{XY}(x,y) \cdot \log[P_{XY}(x,y) - P_X(x)P_Y(y)]$$
- $\mathrm{Cov}(X;Y) = \iint_{x,y} xy \cdot [P_{XY}(x,y) - P_X(x)P_Y(y)]$
- Electrode values as probability distribution
- ADHD: 2231, CONTROL: 1757, Total: 3988



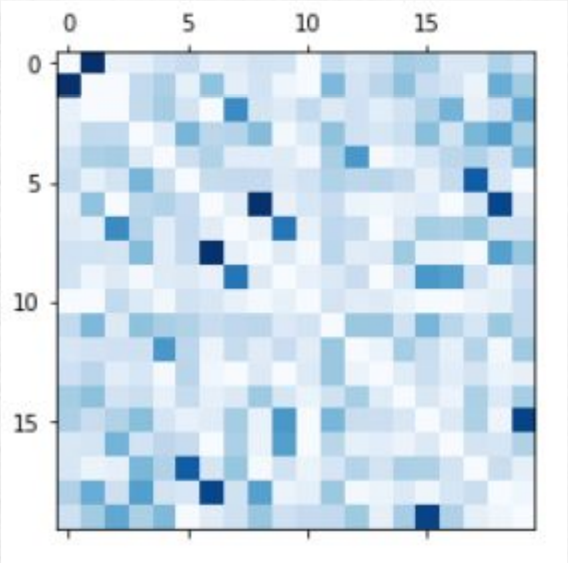
Create Mutual Information Table





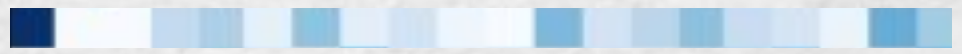
Create Graph Dataset

- Node: 20
- Edges: 190 + 20 (self-loops)
- Data: $|V| \times |F|$ (20 nodes x 20 features)
- Label: {0,1}
- Unweighted
- Undirected
- Total graphs: 3988 (number of MI tables)



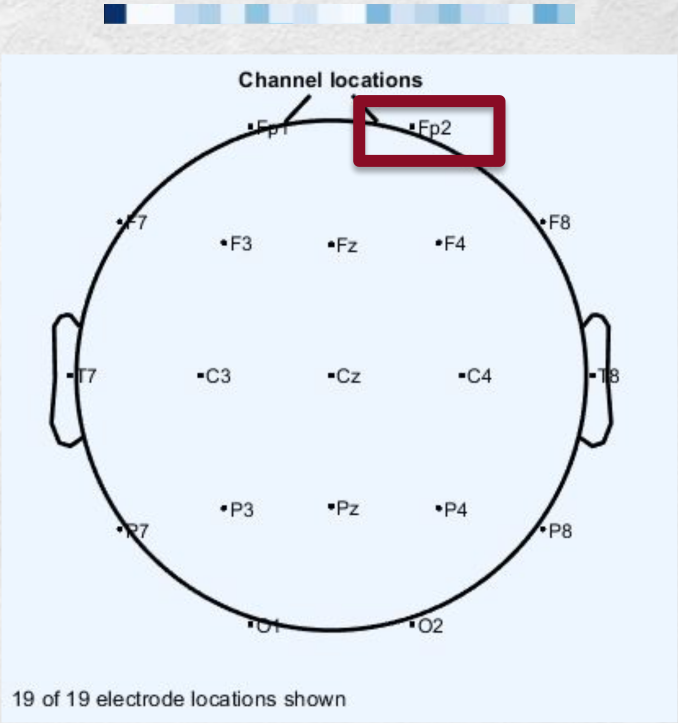
Create Graph Dataset

- Node: 20
- Edges: 190 + 20 (self-loops)
- Data: $|V| \times |F|$ (20 nodes x 20 features)
- Label: $\{0,1\}$
- Unweighted
- Undirected
- Total graphs: 3988 (number of MI tables)



Create Graph Dataset

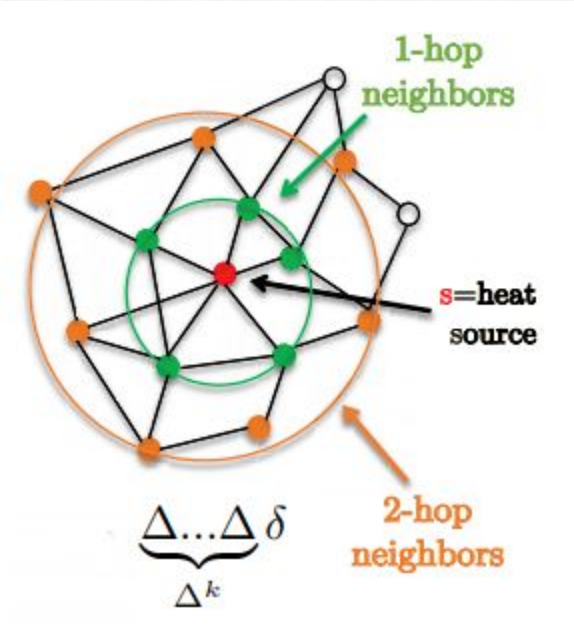
- Node: 20
- Edges: 190 + 20 (self-loops)
- Data: $|V| \times |F|$ (20 nodes x 20 features)
- Label: {0,1}
- Unweighted
- Undirected
- Total graphs: 3988 (number of MI tables)



GNN Model

- K-hop neighbors
- Aggregate Function

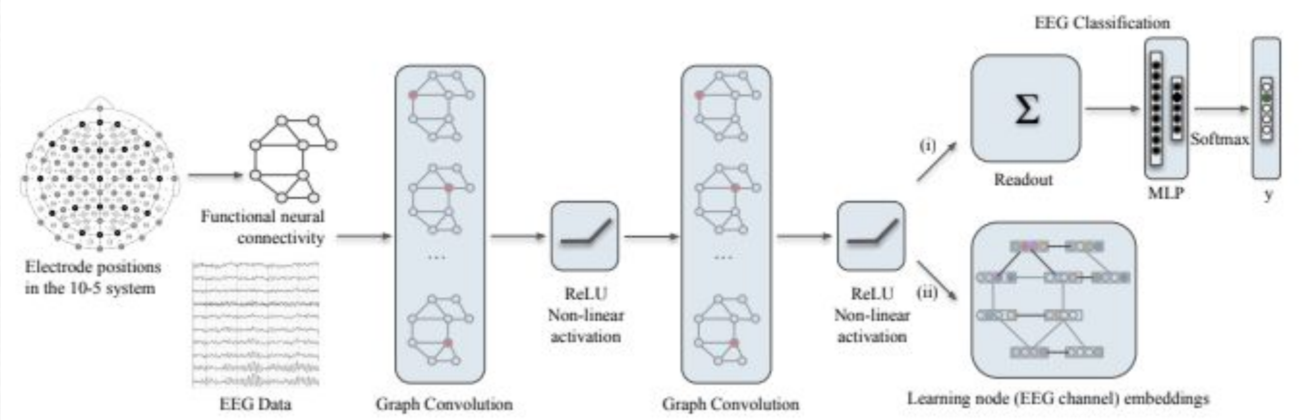
  For each node feature vector
  - Find mean of features of n-hop neighbors
    - Concatenate n^{th} ,$(n-1)^{th}$ node representation (GraphSAGE)
    - Weight matrix of size $(F' \times F)$
      - $W \cdot h : (F', F) \times (F, 2)$
    - Non-linearity function (ReLU)
    - Normalize
- Graph representation
  - Take mean over all nodes (element-wise)
  - Same dimension as single node



GNN Model

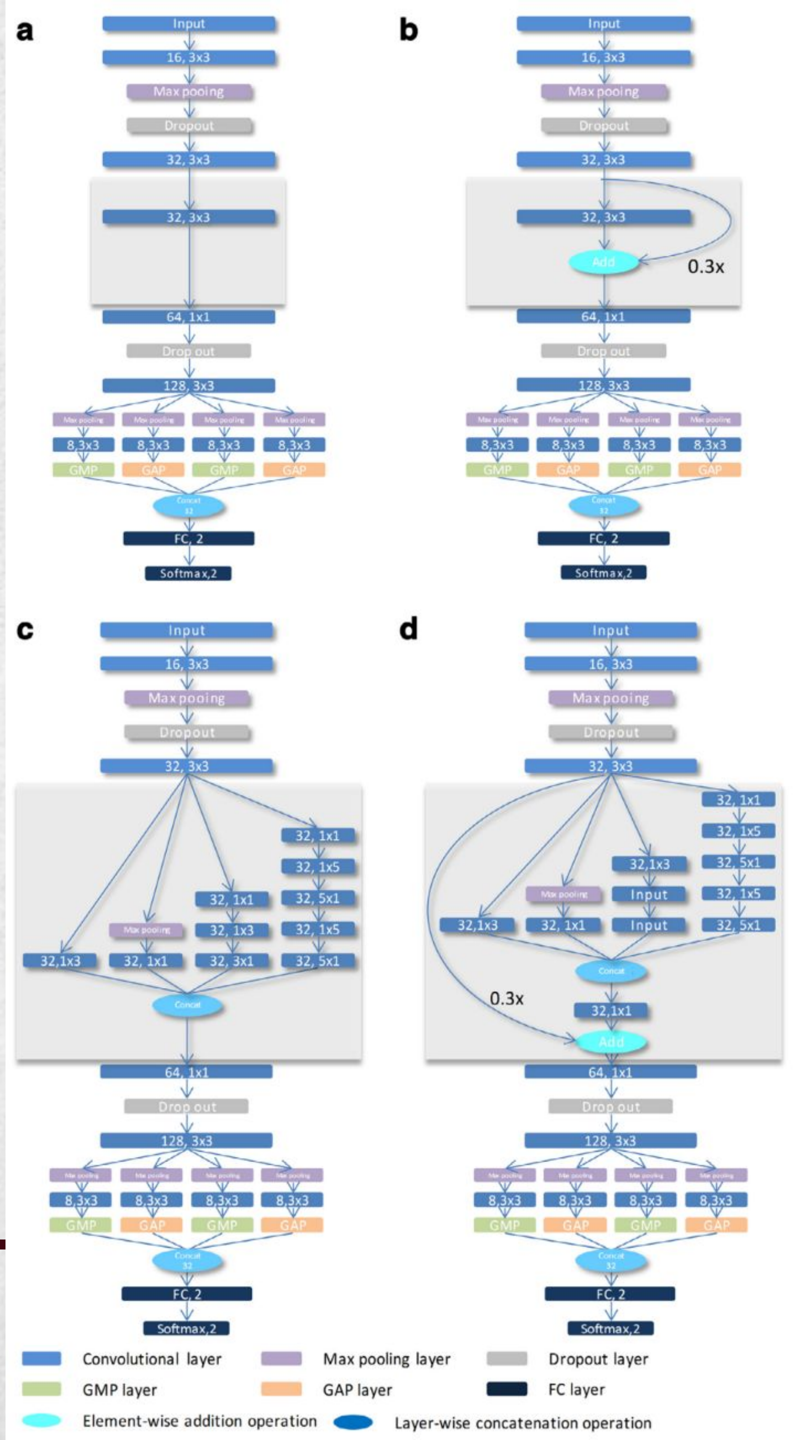
- K-hop neighbors
- Aggregate Function

  For each node feature vector
  - Find mean of features of n-hop neighbors
    - Concatenate n^{th} ,$(n-1)^{th}$ node representation (GraphSAGE)
    - Weight matrix of size $(F' \times F)$
      - $W \cdot h : (F', F) \times (F, 2)$
    - Non-linearity function (ReLU)
    - Normalize
- Graph representation
  - Take mean over all nodes (element-wise)
  - Same dimension as single node

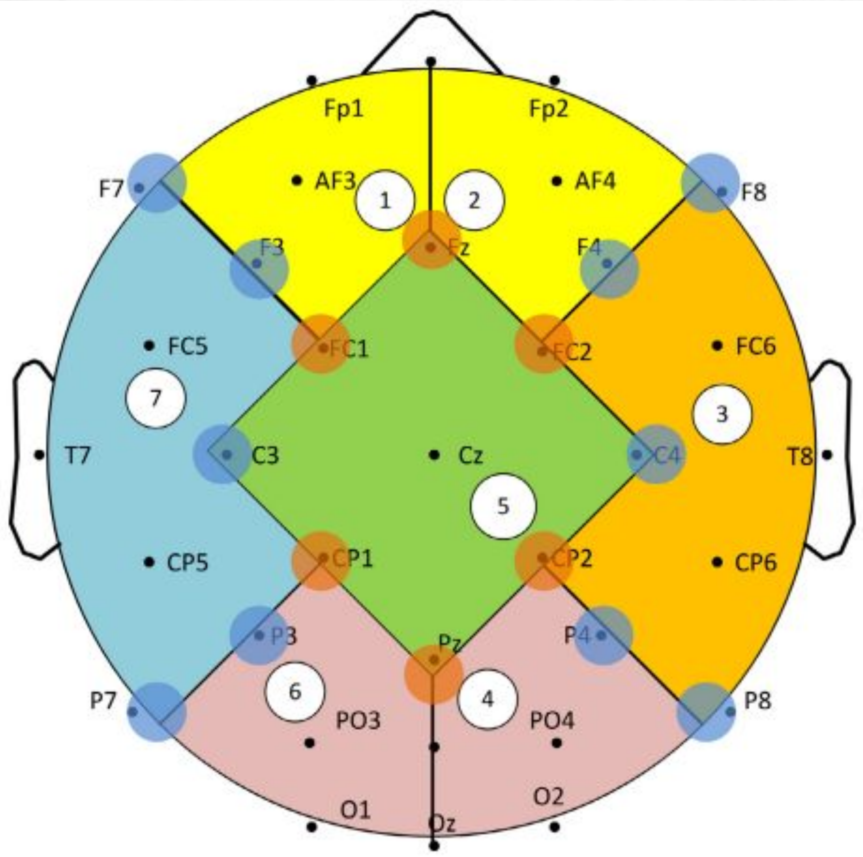


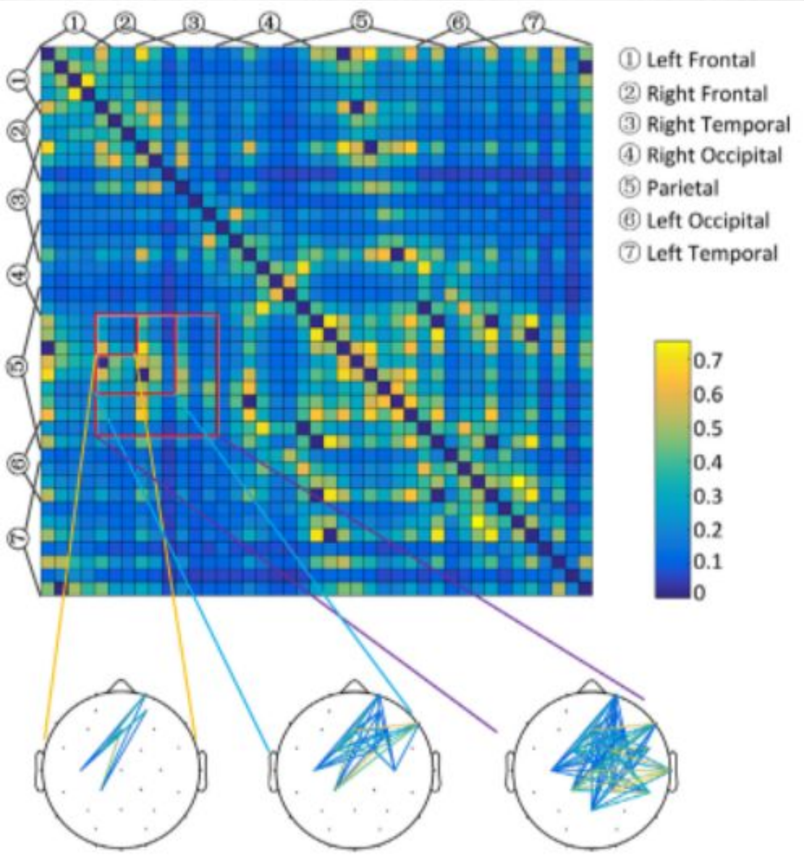
Replicate CNN

- Chen et al. used a different dataset.
  - Number of patients
  - Number of electrodes
- Best performing model from 4 proposed models

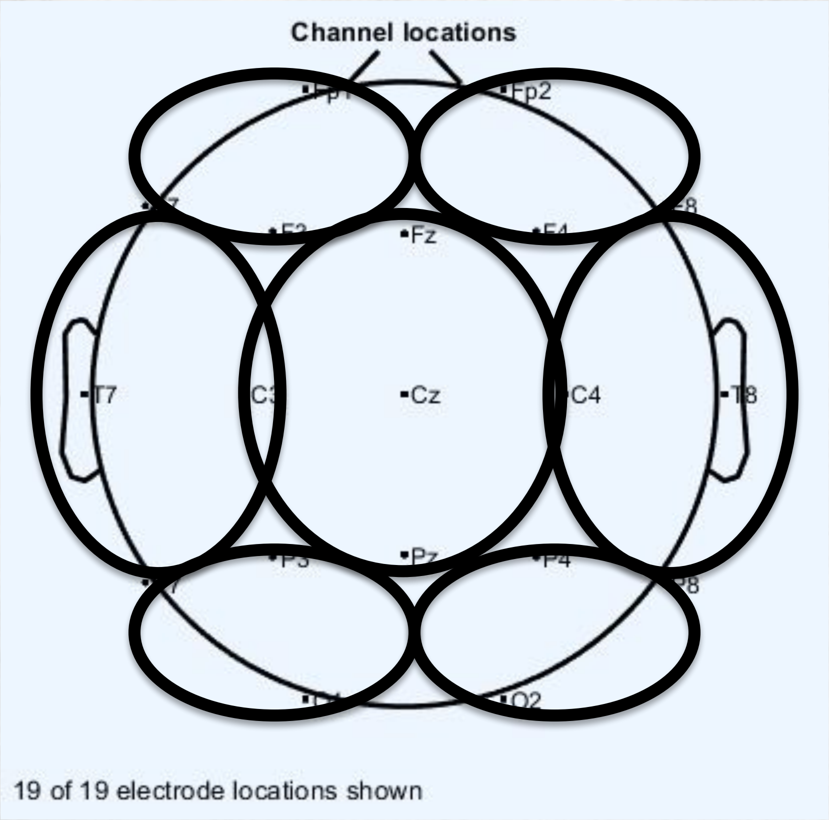


Replicate CNN



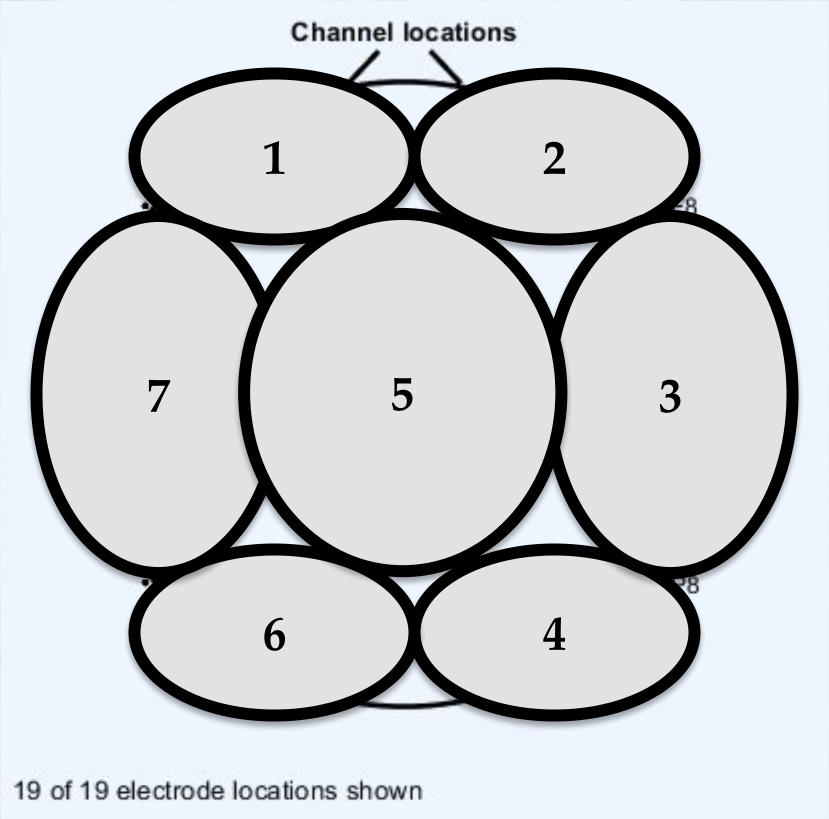


Replicate CNN



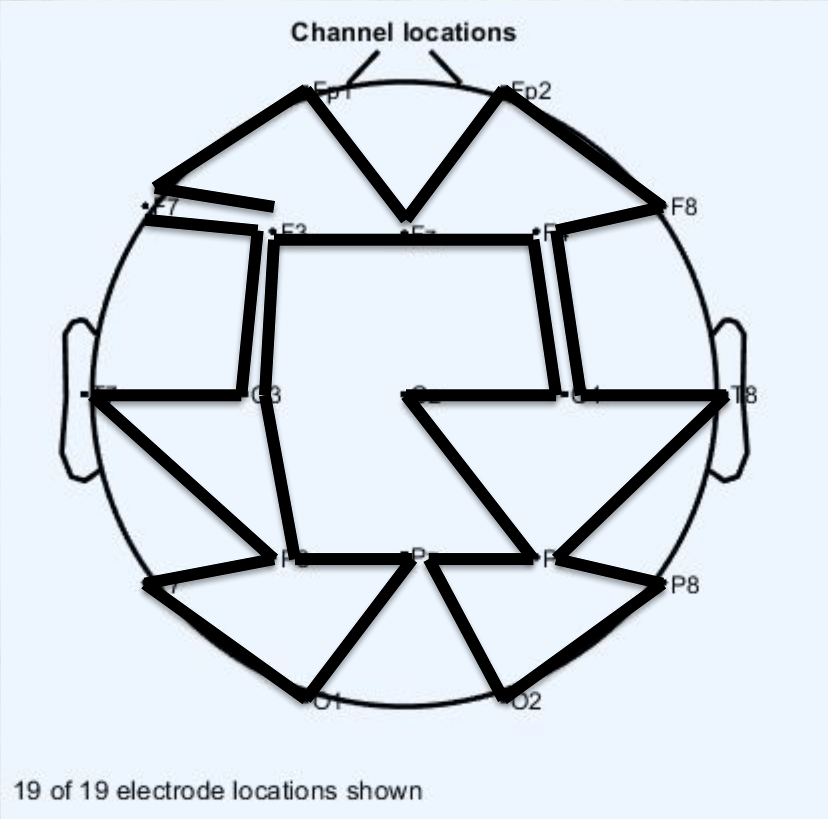
- Group Electrodes

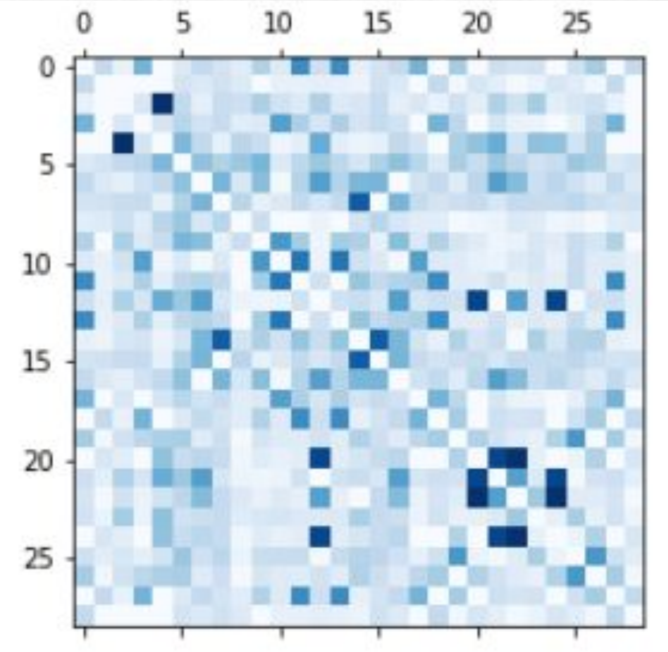
Replicate CNN



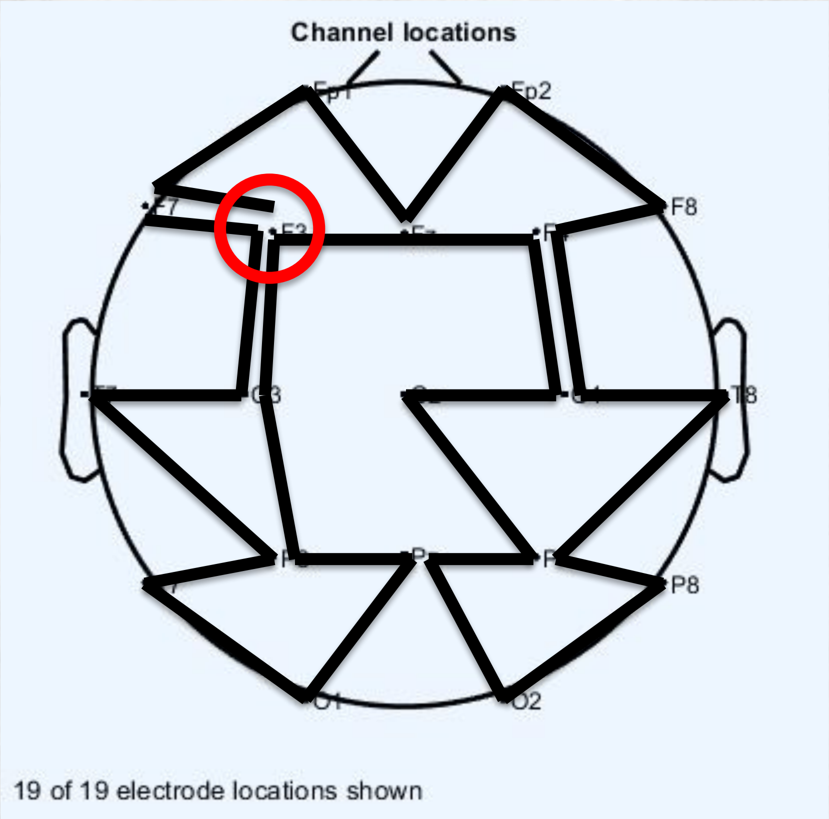
1. Left Frontal
2. Right Frontal
3. Right Temporal
4. Parietal
5. Left Occipital
6. Left Temporal

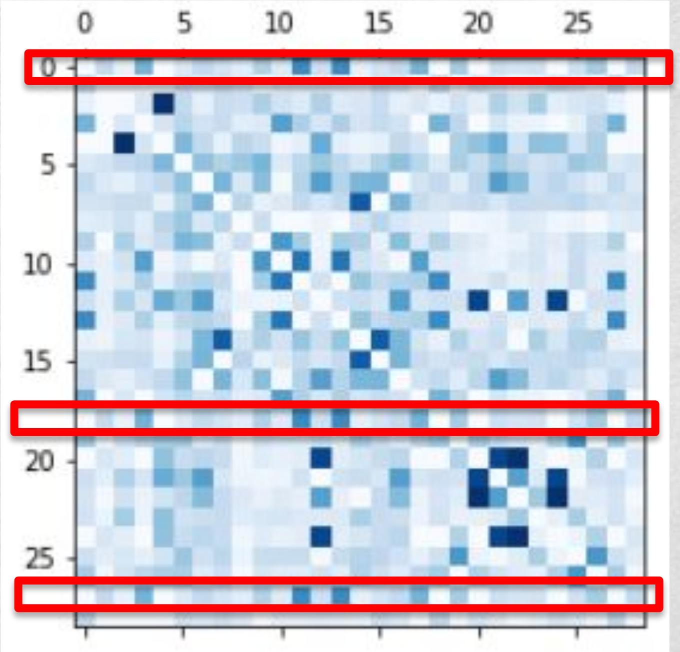
Replicate CNN



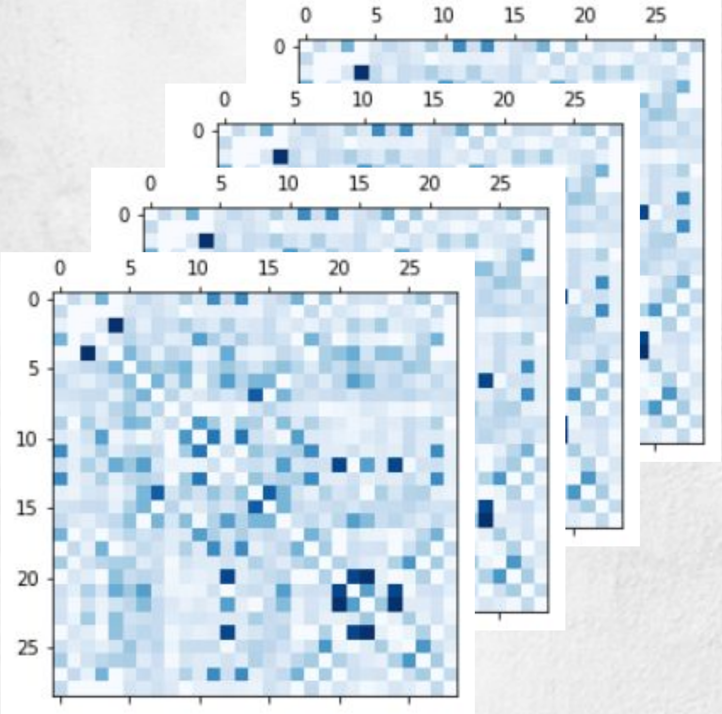


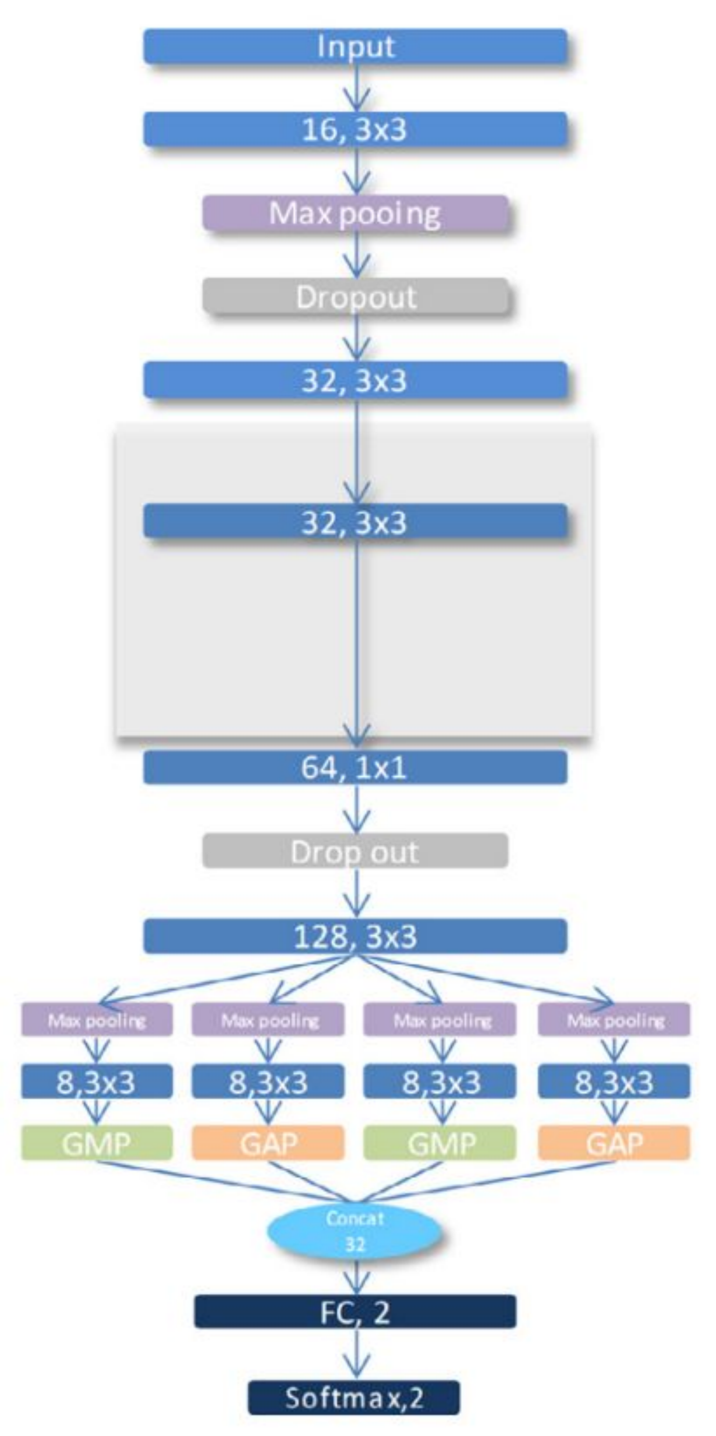
Replicate CNN





Replicate CNN



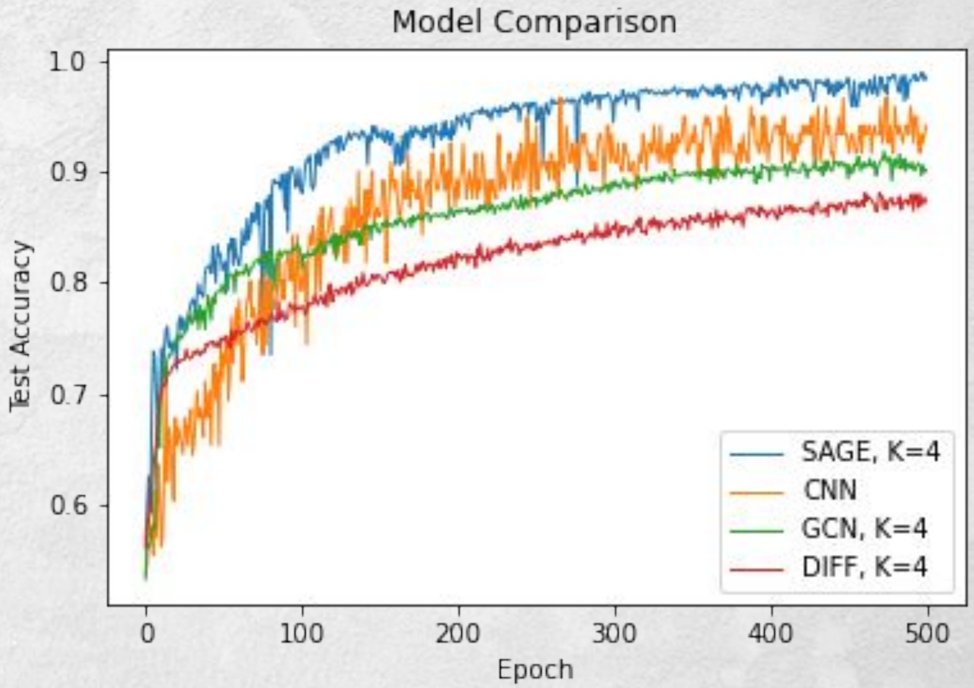


Conclusion

DIFF				
K-hop	1	2	3	4
Train	81.09	81.98	85.59	86.35
Validation	80.15	84.67	85.46	89.47
Test	81.59	82.85	85.68	87.14
GCN				
K-hop	1	2	3	4
Train	76.91	82.81	90.28	90.84
Validation	76.13	84.42	88.44	89.95
Test	75.53	81.92	89.92	90.25
SAGE				
K-hop	1	2	3	4
Train	82.26	95.52	98.22	98.86
Validation	82.91	92.46	96.48	96.98
Test	82.05	95.46	97.97	98.65
CNN				
Train	99.33			
Validation	100			
Test	94.21			

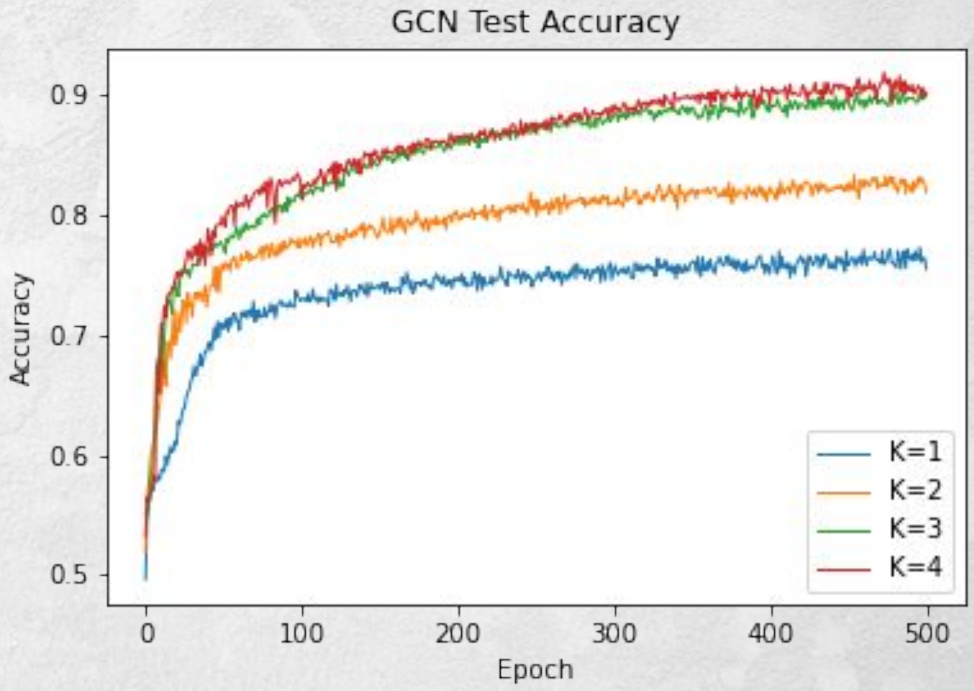
Conclusion

DIFF				
K-hop	1	2	3	4
Train	81.09	81.98	85.59	86.35
Validation	80.15	84.67	85.46	89.47
Test	81.59	82.85	85.68	$\boxed{87.14}$
GCN				
K-hop	1	2	3	4
Train	76.91	82.81	90.28	90.84
Validation	76.13	84.42	88.44	89.95
Test	75.53	81.92	89.92	$\boxed{90.25}$
SAGE				
K-hop	1	2	3	4
Train	82.26	95.52	98.22	98.86
Validation	82.91	92.46	96.48	96.98
Test	82.05	95.46	97.97	$\boxed{98.65}$
CNN				
Train	99.33			
Validation	100			
Test	$\boxed{94.21}$			



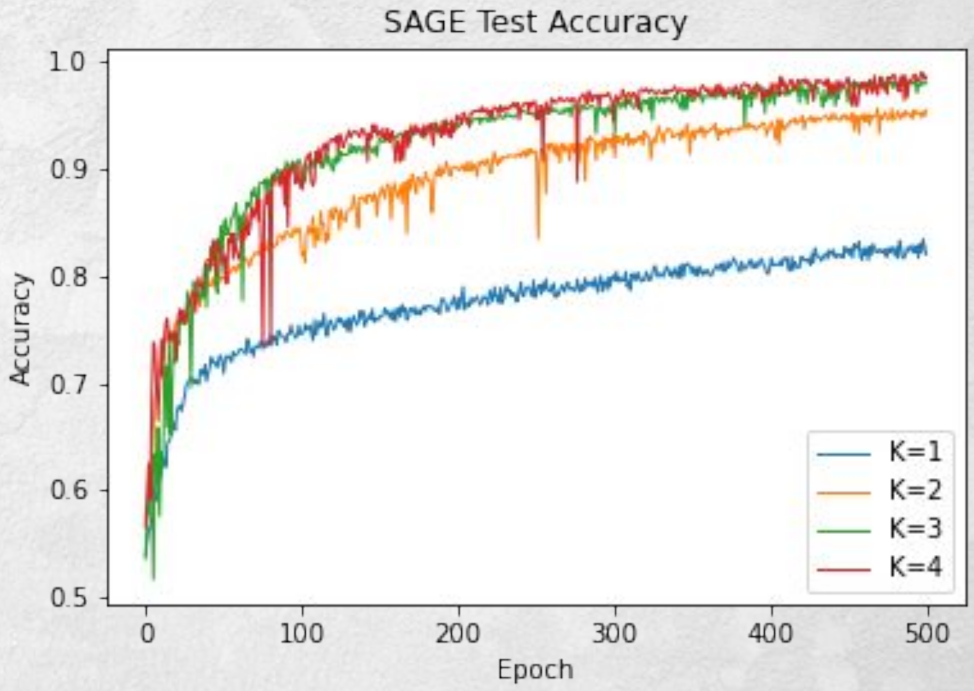
Conclusion

DIFF				
K-hop	1	2	3	4
Train	81.09	81.98	85.59	86.35
Validation	80.15	84.67	85.46	89.47
Test	81.59	82.85	85.68	87.14
GCN				
K-hop	1	2	3	4
Train	76.91	82.81	90.28	90.84
Validation	76.13	84.42	88.44	89.95
Test	75.53	81.92	89.92	90.25
SAGE				
K-hop	1	2	3	4
Train	82.26	95.52	98.22	98.86
Validation	82.91	92.46	96.48	96.98
Test	82.05	95.46	97.97	98.65
CNN				
Train	99.33			
Validation	100			
Test	94.21			



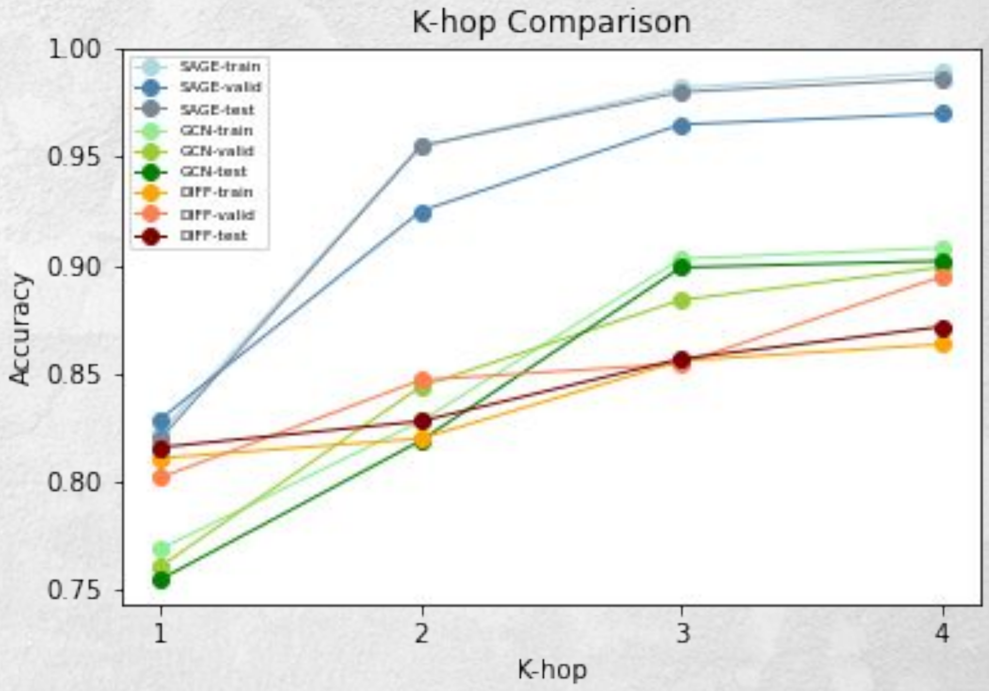
Conclusion

DIFF				
K-hop	1	2	3	4
Train	81.09	81.98	85.59	86.35
Validation	80.15	84.67	85.46	89.47
Test	81.59	82.85	85.68	87.14
GCN				
K-hop	1	2	3	4
Train	76.91	82.81	90.28	90.84
Validation	76.13	84.42	88.44	89.95
Test	75.53	81.92	89.92	90.25
SAGE				
K-hop	1	2	3	4
Train	82.26	95.52	98.22	98.86
Validation	82.91	92.46	96.48	96.98
Test	82.05	95.46	97.97	98.65
CNN				
Train	99.33			
Validation	100			
Test	94.21			



Conclusion

DIFF				
K-hop	1	2	3	4
Train	81.09	81.98	85.59	86.35
Validation	80.15	84.67	85.46	89.47
Test	81.59	82.85	85.68	87.14
GCN				
K-hop	1	2	3	4
Train	76.91	82.81	90.28	90.84
Validation	76.13	84.42	88.44	89.95
Test	75.53	81.92	89.92	90.25
SAGE				
K-hop	1	2	3	4
Train	82.26	95.52	98.22	98.86
Validation	82.91	92.46	96.48	96.98
Test	82.05	95.46	97.97	98.65
CNN				
Train	99.33			
Validation	100			
Test	94.21			



Conclusion

- GCN, DIFF did not outperform CNN.
  - "The majority of these methods do not scale to large graphs or are designed for whole-graph classification." (Hamilton et al.)
- SAGE outperformed CNN.
  - More suitable for brain-signal research
- Accuracy Increased for bigger K value
  - Brain stimulation
  - Needs to be compared with neuroscience literature

Further Research Direction

- Other methods to construct adjacency table.
  - Pearson Correlation, Dynamic Time Warping.
- Other GNN models
  - Graph Isomorphism Network, BrainGNN, SGCN.
- Compare with literature on ADHD research in neuroscience.
- Try bigger values of k.
- Interpretable Machine Learning
  - Sensitivity analysis
    - Node-level, graph-level features for GNN model training
  - Omit certain channels when training

Thank You

Questions / Suggestions