KAGGLE

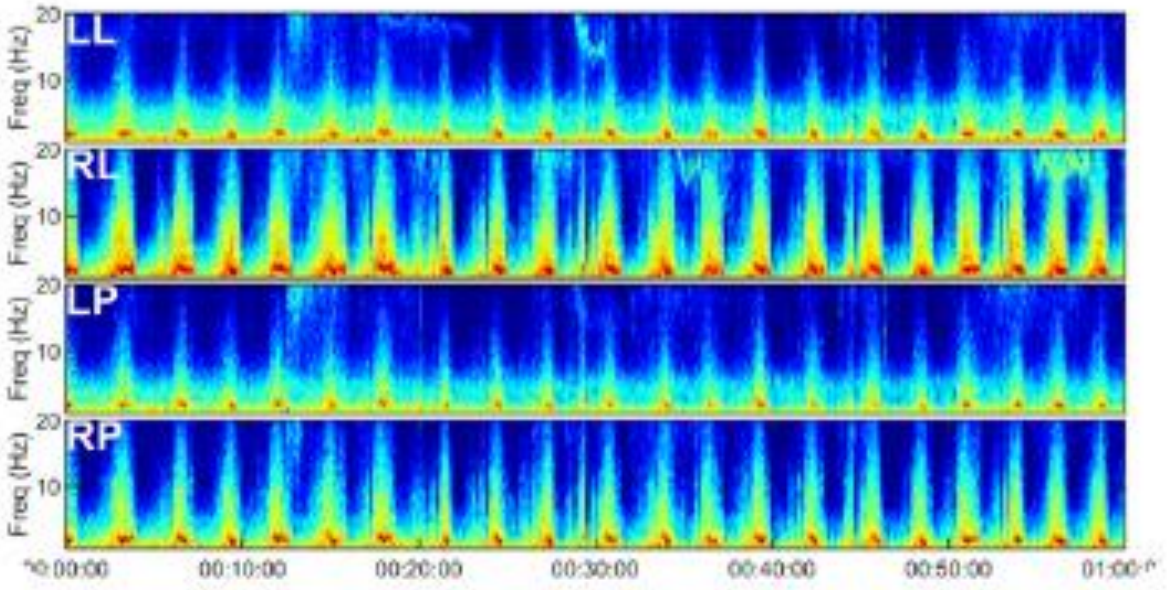
HMS

\EMILY CHI

AGENDA

- Background
- Data
- Modeling
- Training methods
- Overall code review
- Important findings


BACKGROUND



HARVARD MEDICAL SCHOOL · RESEARCH CODE COMPETITION · A DAY AGO

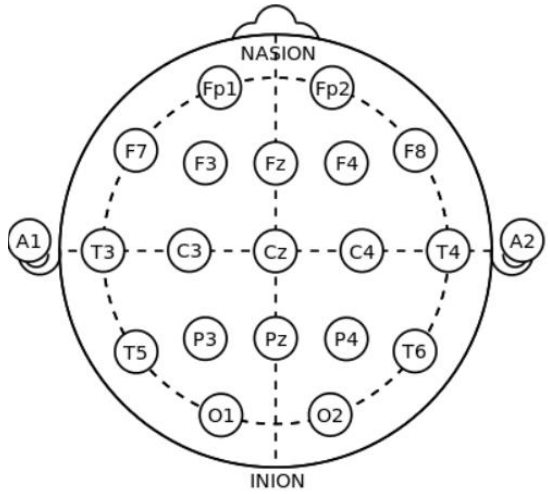
HMS - Harmful Brain Activity Classification

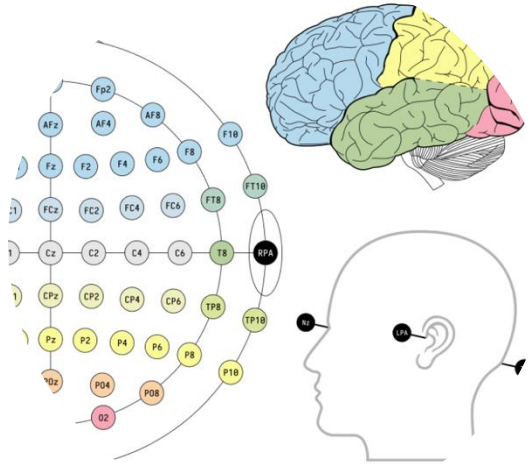
Classify seizures and other patterns of harmful brain activity in critically ill patients

THE GOAL OF THIS COMPETITION IS TO DETECT AND CLASSIFY SEIZURES AND OTHER TYPES OF HARMFUL BRAIN ACTIVITY.

EEGS 10–20 SYSTEM

- In this system, electrode locations are named after the brain regions below them.
- These letters may be accompanied by numbers indicating the electrode position relative to the center line of the head (odd for the left, even for the right, z for the electrode on the center line).

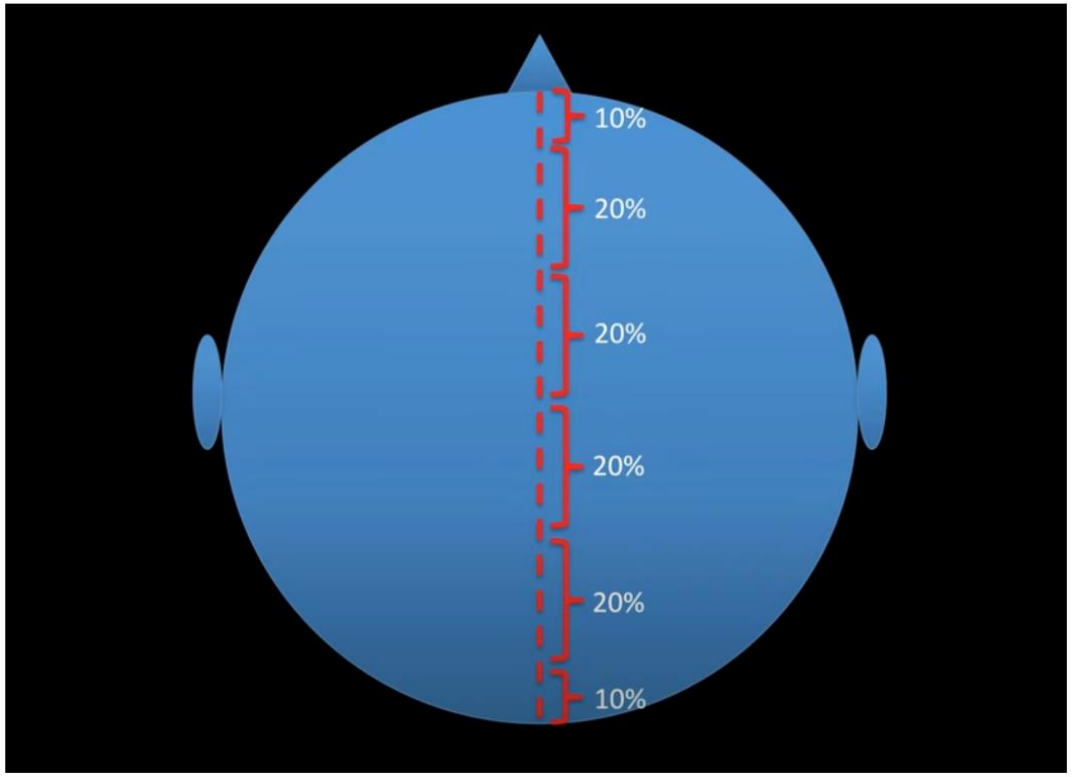


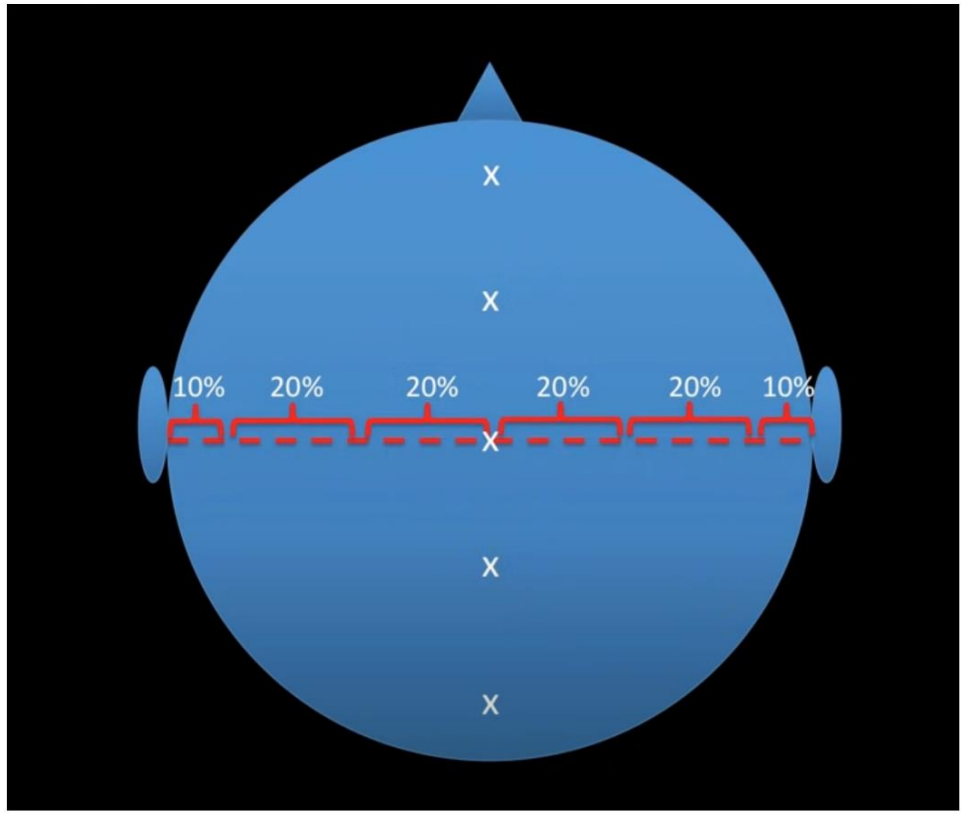


%28EEG%29

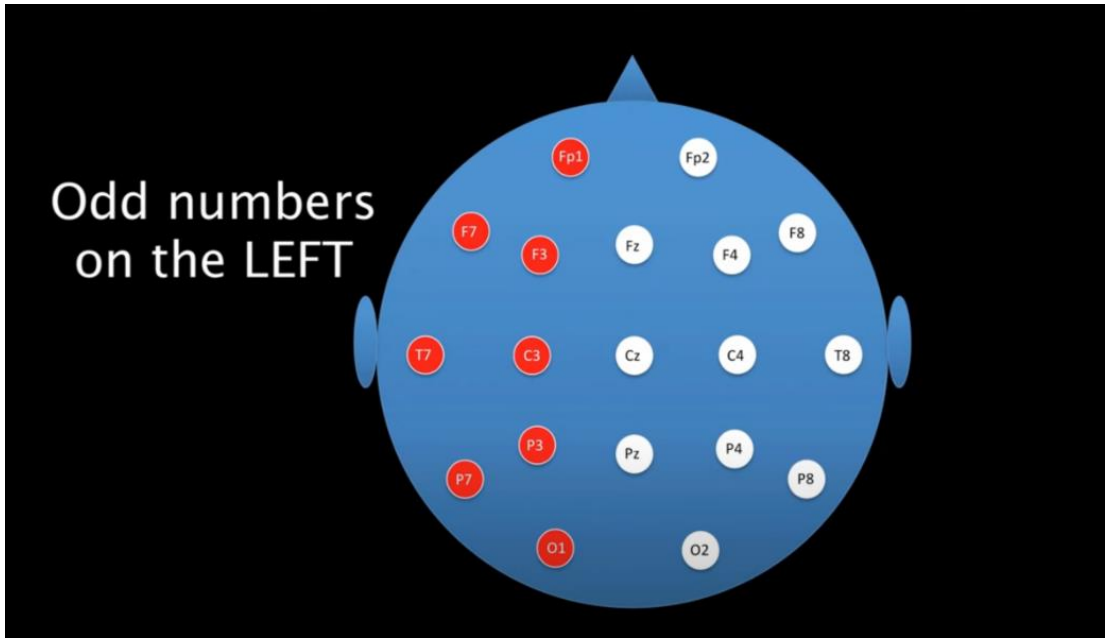
EEGS 10–20 SYSTEM

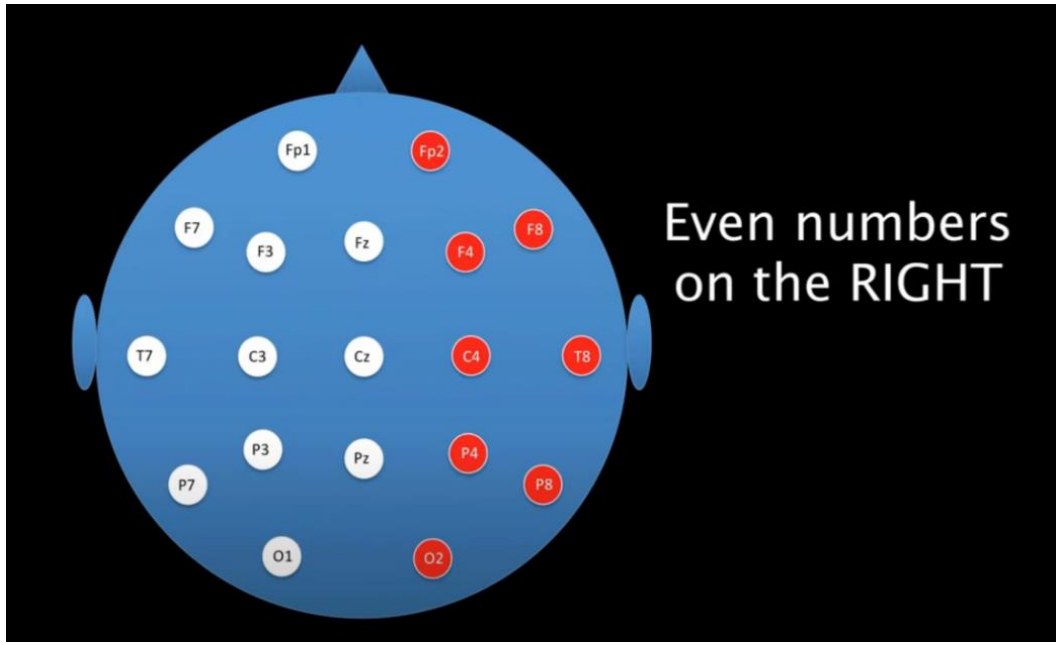
Why it called 10-20 system ?





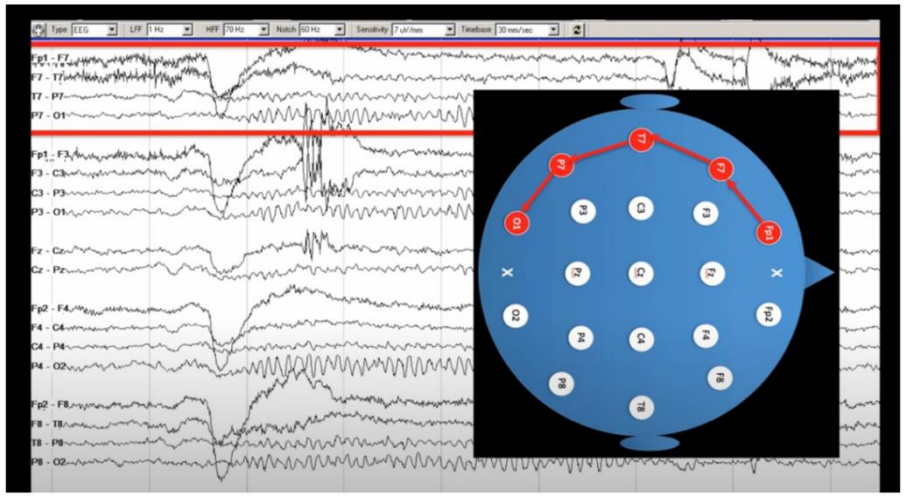
EEGS 10–20 SYSTEM



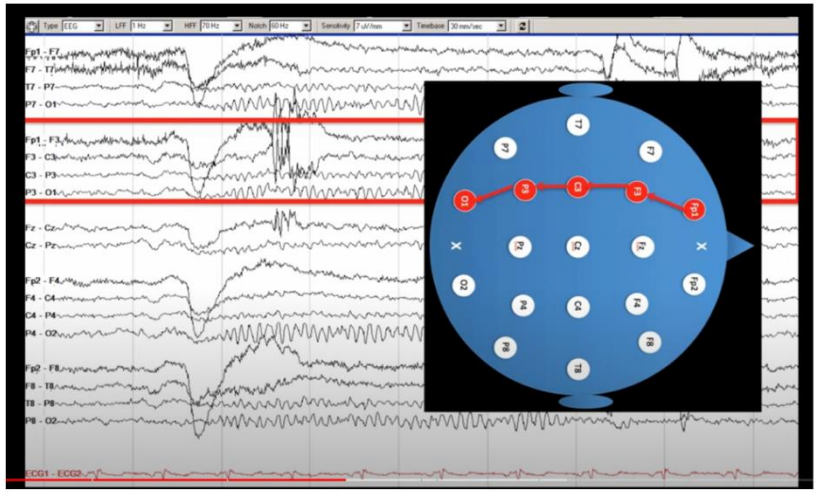


EEGS 10–20 SYSTEM

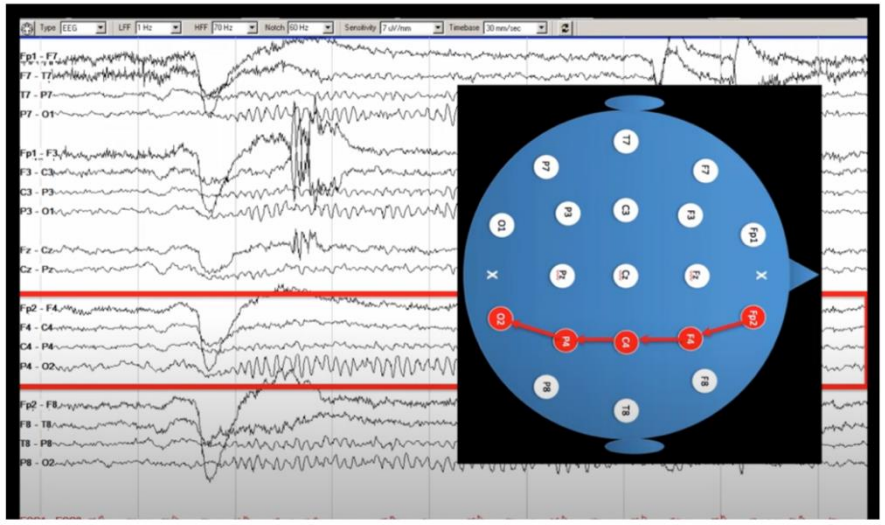
LL



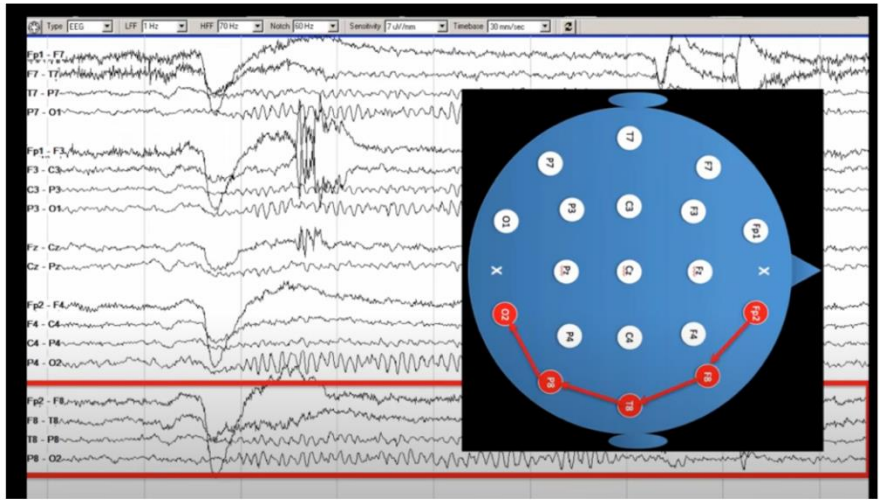
LP



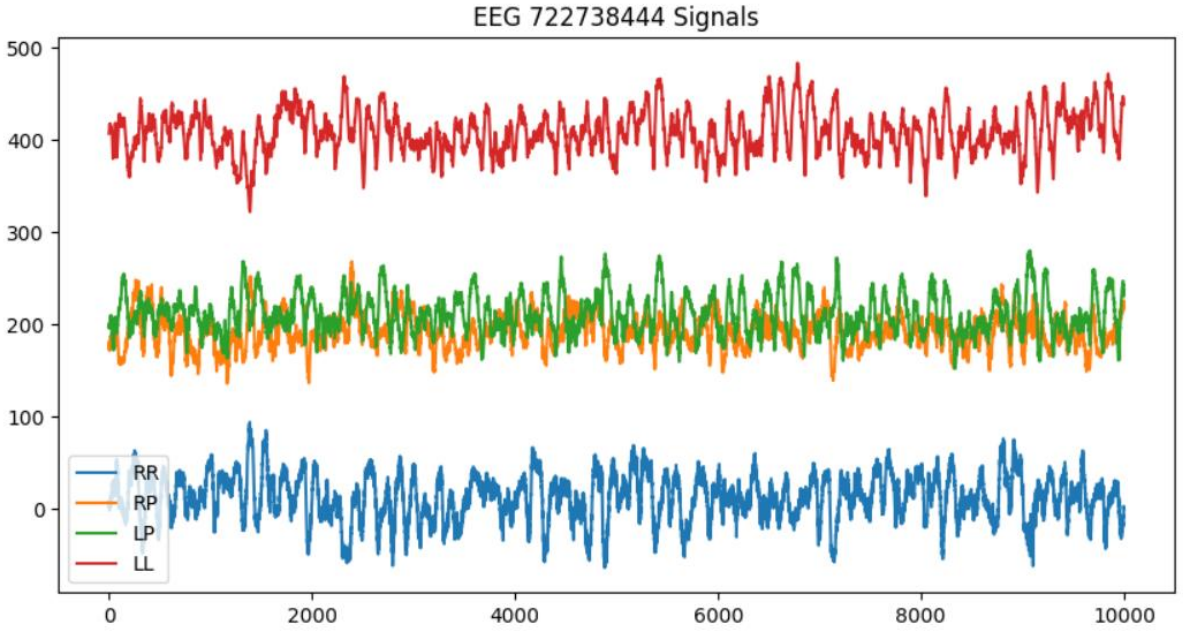
RP



RR



HOW TO CREATE SPECTROGRAM FROM EEG?



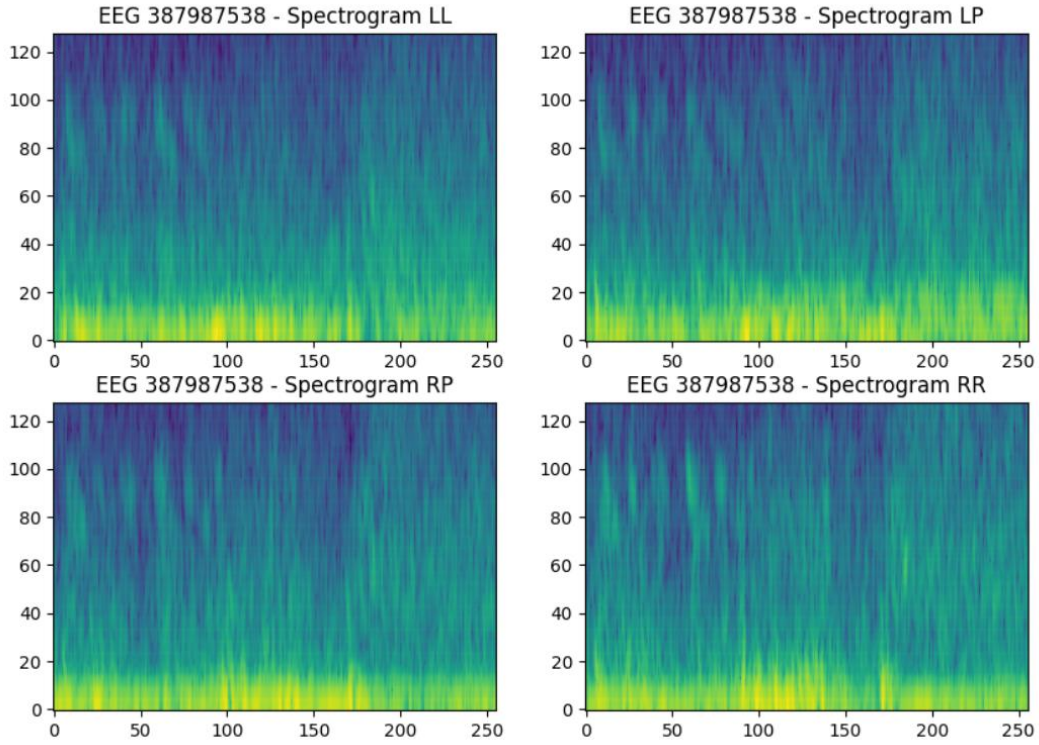
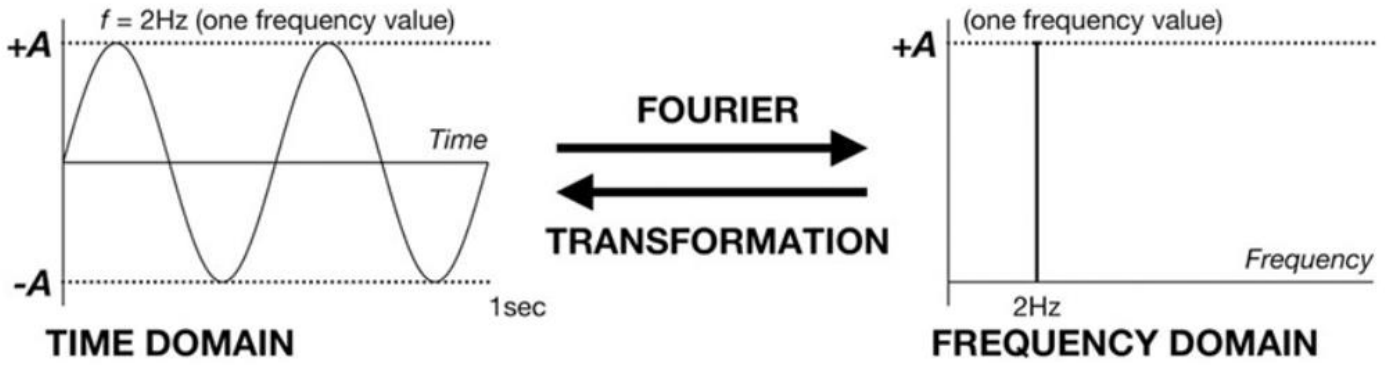
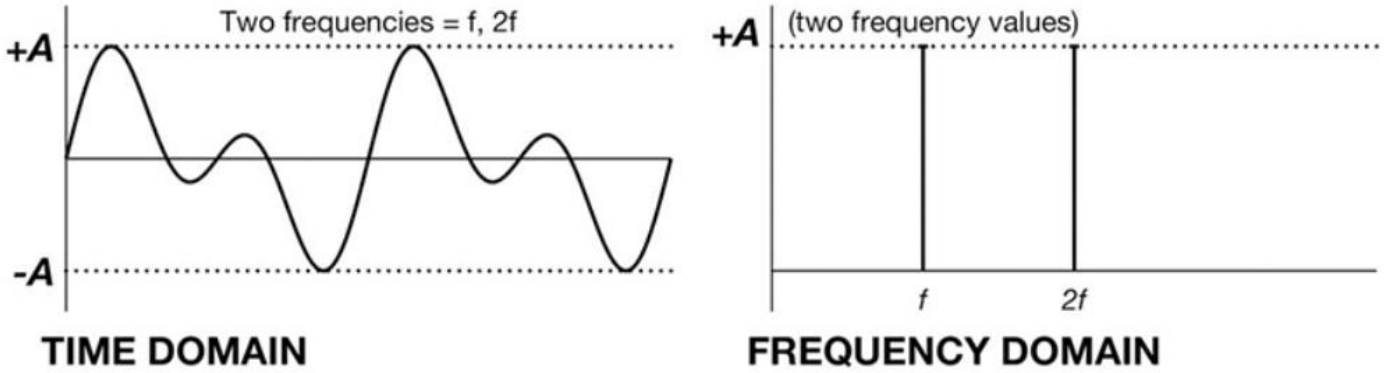


FIG. 3.



Schematic of Fourier transformation.

FIG. 4.



An example of mixing oscillations in the time and frequency domains.

SPECTROGRAMS

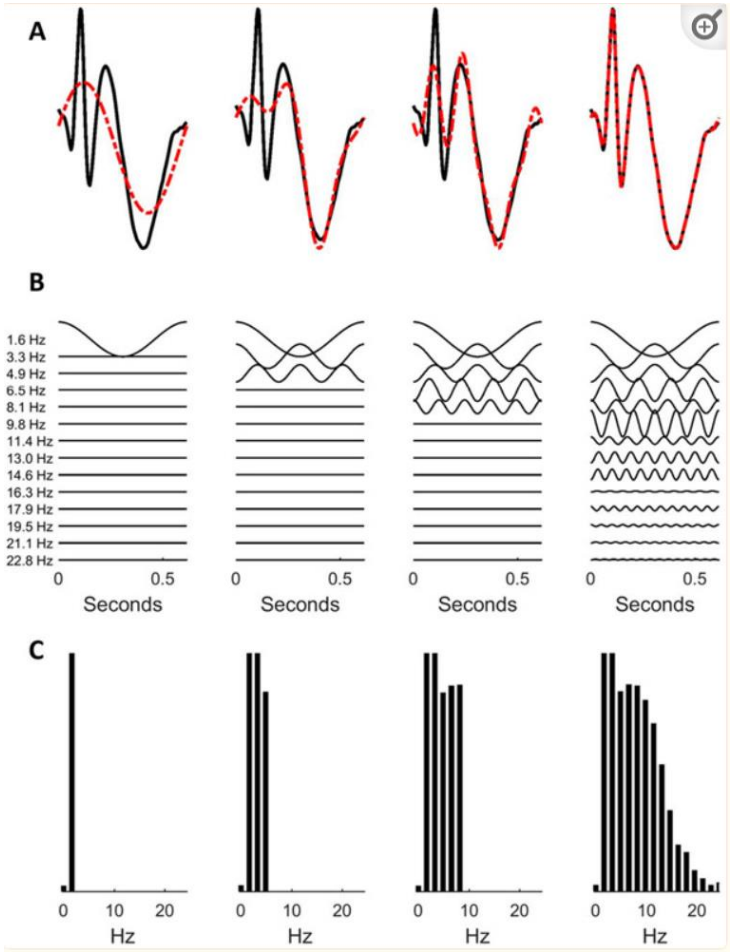
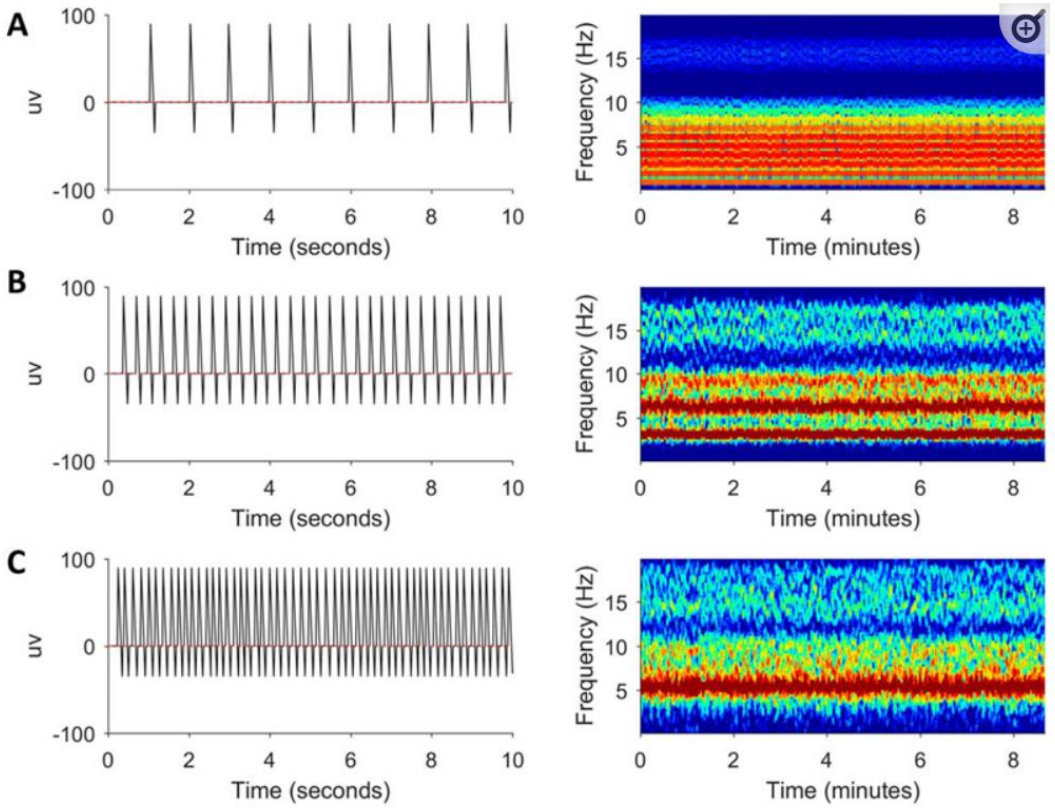


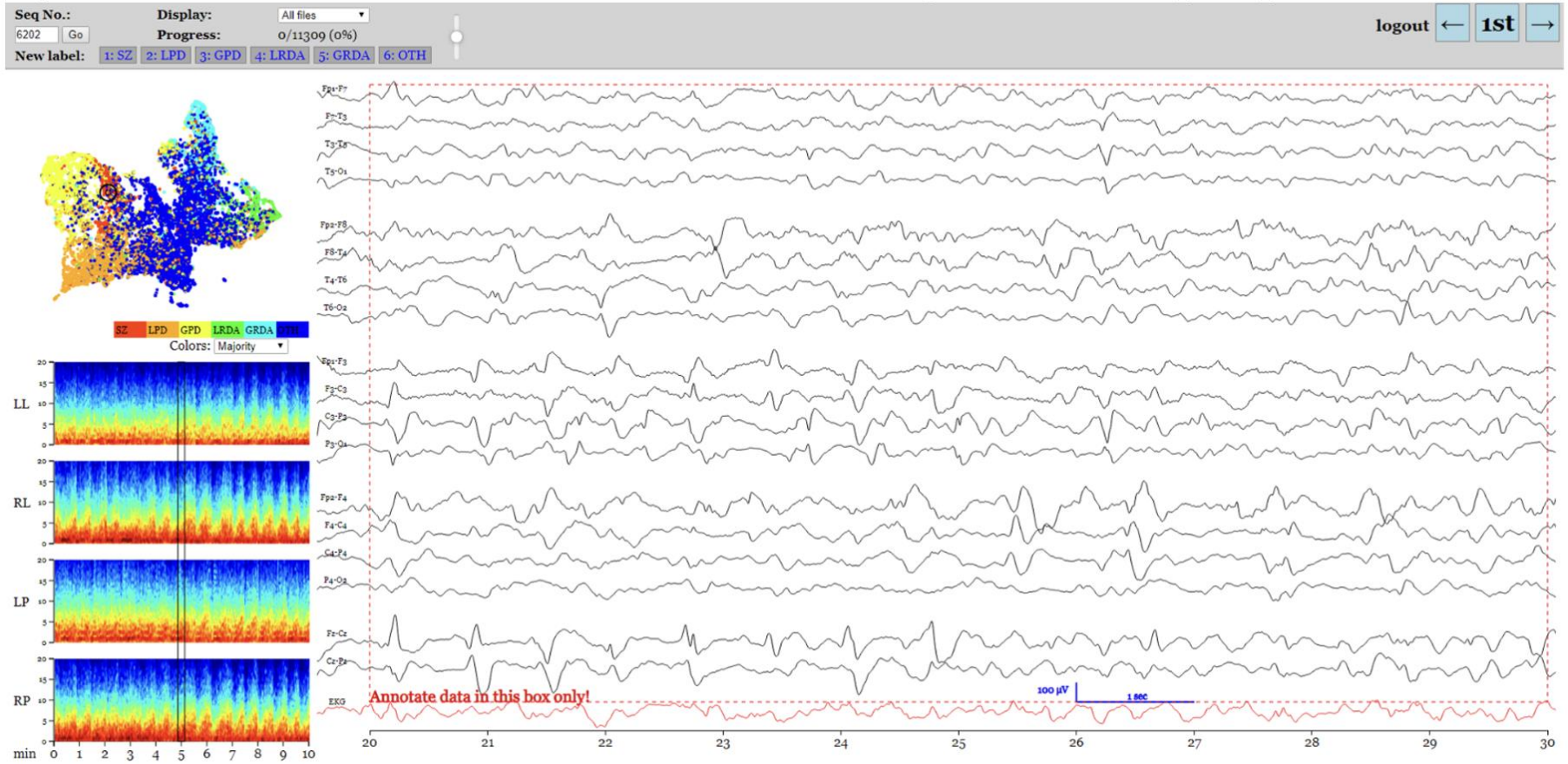
FIG. 8.



Simulated periodic discharges at 1 (**A**), 3 (**B**), and 5 Hz (**C**) with corresponding spectrograms.

KAGGLE DATA

Figure S1. Web-based GUI used to collect annotations of EEG segments from multiple experts.




DATA

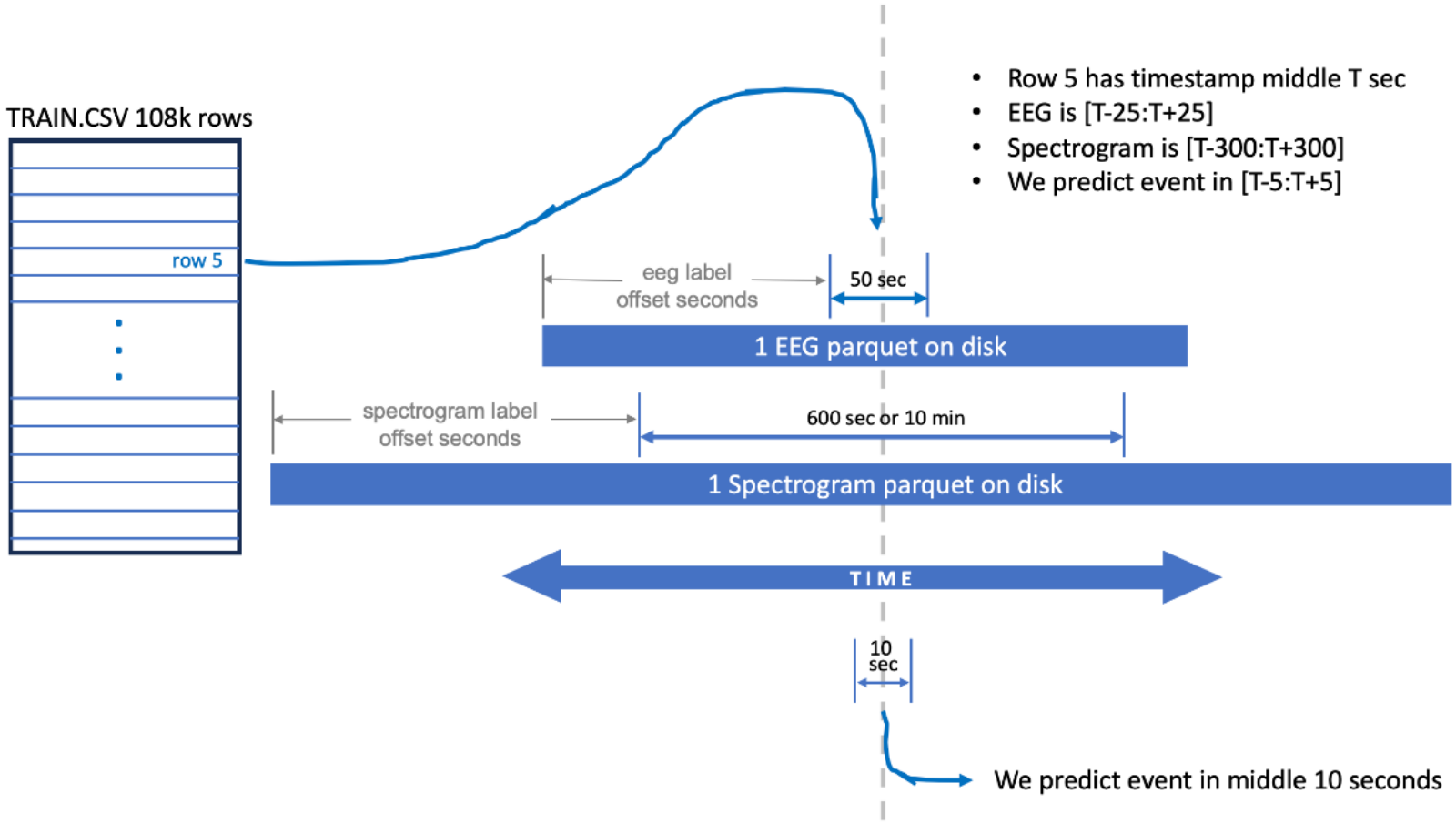
train.csv - Training set metadata. Expert reviewers examined 50-second EEG samples and matching spectrograms centered on the same time with 10-minute windows, labeling the central 10 seconds. Many such samples overlap and have been merged. train.csv provides metadata that you can use to extract the annotated original subset.

- eeg_id - Unique identifier for the complete EEG recording
- eeg_sub_id - ID for the specific 50-second sub-sample this label applies to
- eeg_label_offset_seconds - Time from the start of the merged EEG to this sub-sample
- spectrogram_id - Unique identifier for the complete EEG recording
- spectrogram_sub_id - ID for the specific 10-minute sub-sample this label applies to
- spectogram_label_offset_seconds - Time from the start of the merged spectrogram to this sub-sample
- label_id - ID for a set of labels
- patient_id - ID of the patient providing the data
- expert_consensus - Consensus label from reviewers. Provided for convenience only.
- [seizure/lpd/gpd/lrda/grda/other]_vote - Number of reviewer votes for each brain activity category. Full names of activity categories are: lpd: Lateral Periodic Discharges, gpd: Generalized Periodic Discharges, lrda: Lateral Rhythmic Delta Activity, grda: Generalized Rhythmic Delta Activity. Detailed explanations of these patterns can be found here.

test.csv - Test set metadata.

Since there are no overlapping samples in the test set, many columns from the training metadata do not apply.

- train.csv contains 106,800 rows of data but only involves 17,089 unique EEG recordings (eeg_ids), 11,138 unique spectrogram recordings (spectrogram_ids), and 1,950 unique patients (patient_ids). This indicates that some EEG recordings and spectrograms were sampled multiple times or covered by different time windows.
- Data Explanation Since each row in train.csv represents a specific time window (corresponding to a specific patient_id), each row has a specific center timestamp (in seconds). For example, row 235 might have a center timestamp T exactly at 19:30:06 on May 3, 2023, which is located at the center of its corresponding EEG time window and spectrogram time window. (Note that this center timestamp is not directly provided in the dataframe).
- The EEG time window is 50 seconds long, while the spectrogram time window is 600 seconds long. Both share the same center timestamp. In this competition, we are asked to predict events that occur in the middle 10 seconds of these two times windows:
- Center time is T
- EEG time range is [T-25:T+25]
- Spectrogram time range is [T-300:T+300]
- We need to predict events occurring in the time range [T-5:T+5]

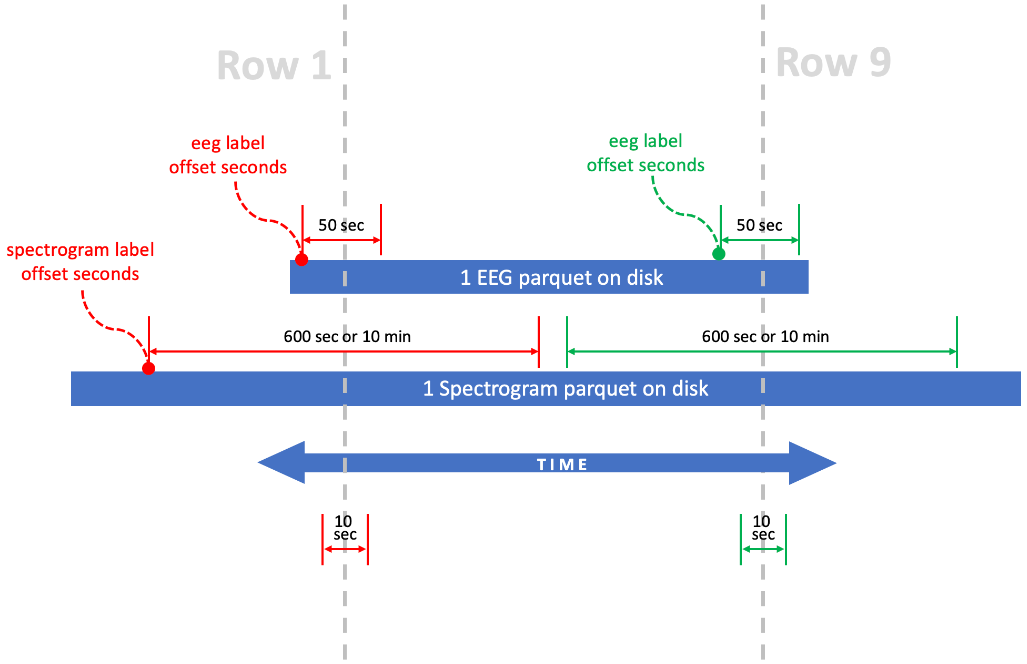

TRAIN.CSV 108k rows
row 5
Row 5 has timestamp middle T sec
EEG is [T-25:T+25]
Spectrogram is [T-300:T+300]
We predict event in [T-5:T+5]
eeg label offset seconds
50 sec
1 EEG parquet on disk
spectrogram label offset seconds
600 sec or 10 min
1 Spectrogram parquet on disk
TIME
10 sec
We predict event in middle 10 seconds

For EEG Parquet files:

- Files contain EEG data spanning more than 50 seconds.
- Each row in the file represents a time window, with each window being 50 seconds long.
- A single EEG Parquet file may contain multiple such time windows, which can be overlapping or consecutive.

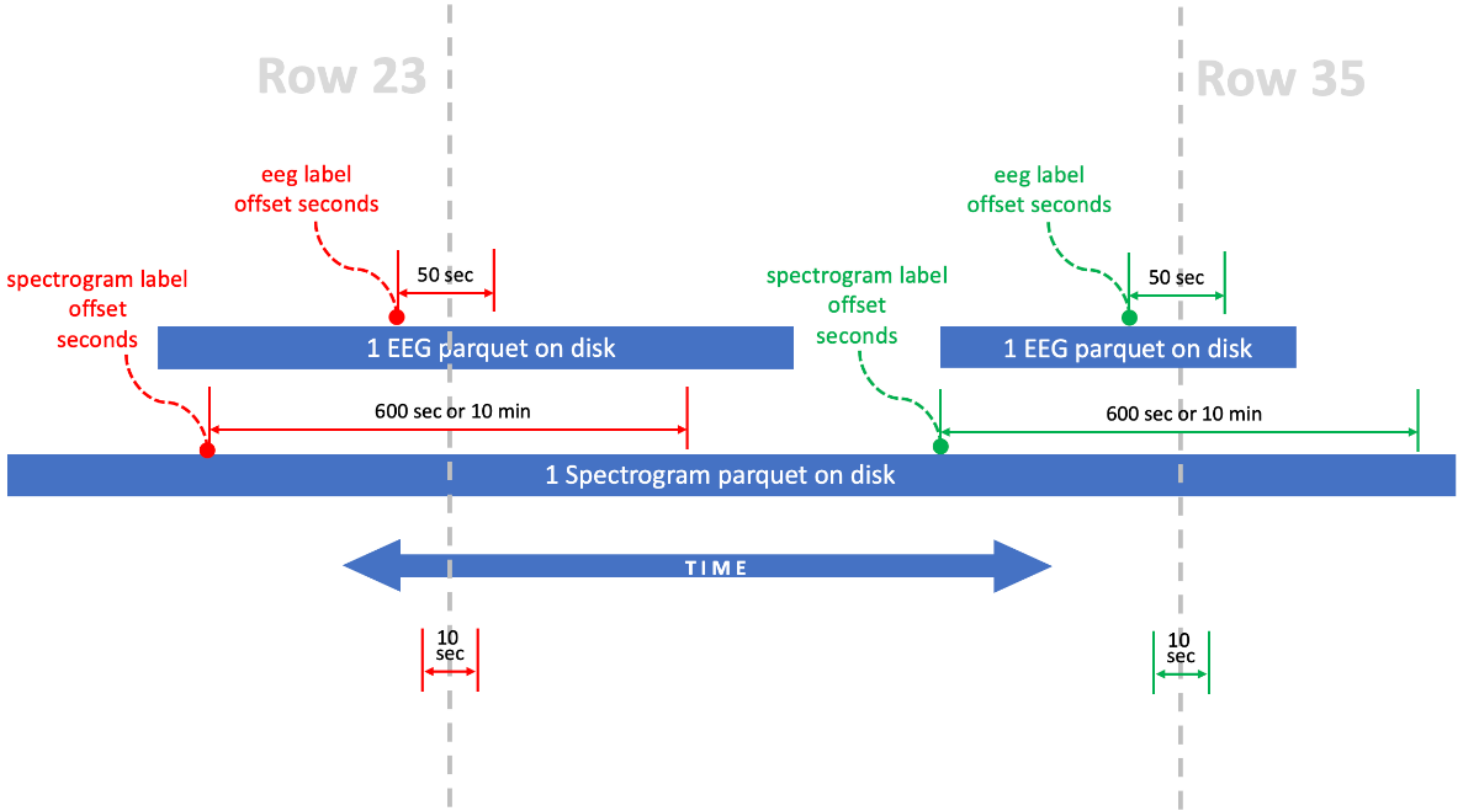
For Spectrogram Parquet files:

- Files contain spectrogram data spanning more than 600 seconds.
- Similarly, each row in the file represents a time window, with each window being 600 seconds long.
- A single spectrogram Parquet file may contain multiple such time windows, which can also be overlapping or consecutive.

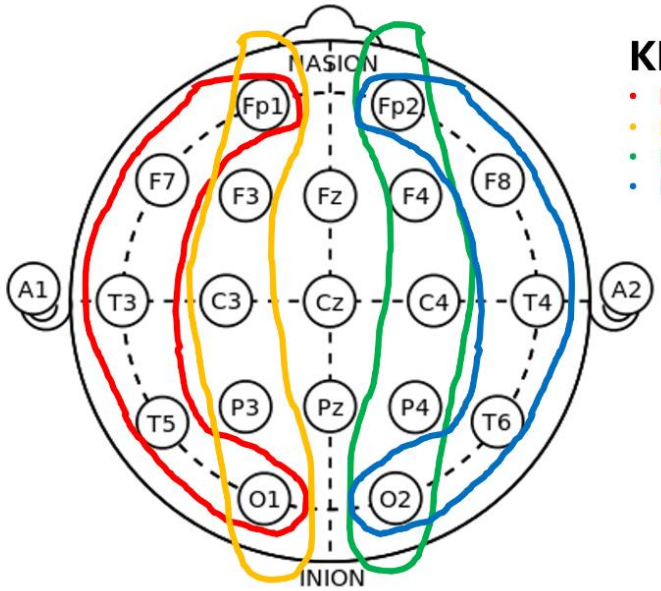


Spectrogram Parquet Files

- Some spectrogram data may cover multiple EEG data windows. Therefore, a specific spectrogram Parquet file may be associated with multiple EEG recordings, which explains why there are fewer spectrogram Parquet files compared to EEG Parquet files.



INPUT DATA-EEGS



KEY

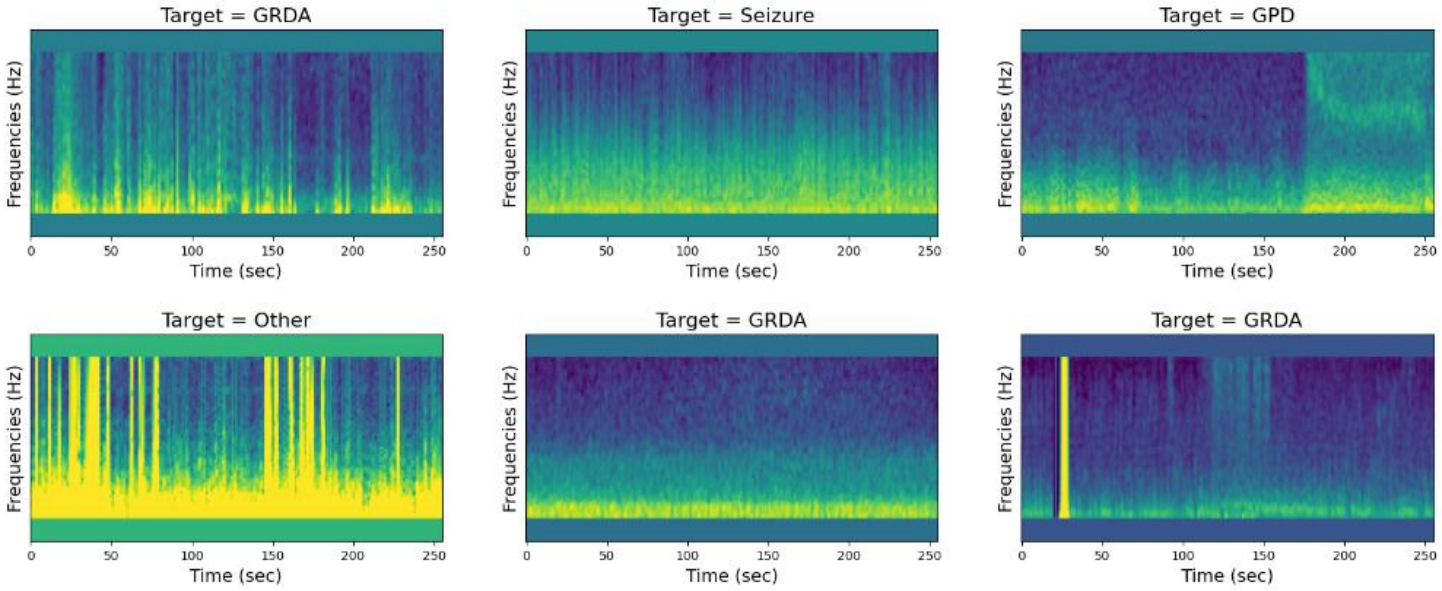
- LL - Left Temporal Chain
- LP - Left Parasagittal Chain
- RP - Right Parasagittal Chain
- RR - Right Temporal Chain

```
# === Feature engineering ===
X[:,0] = data[:,feature_to_index['Fp1']] - data[:,feature_to_index['T3']]
X[:,1] = data[:,feature_to_index['T3']] - data[:,feature_to_index['O1']]

X[:,2] = data[:,feature_to_index['Fp1']] - data[:,feature_to_index['C3']]
X[:,3] = data[:,feature_to_index['C3']] - data[:,feature_to_index['O1']]

X[:,4] = data[:,feature_to_index['Fp2']] - data[:,feature_to_index['C4']]
X[:,5] = data[:,feature_to_index['C4']] - data[:,feature_to_index['O2']]

X[:,6] = data[:,feature_to_index['Fp2']] - data[:,feature_to_index['T4']]
X[:,7] = data[:,feature_to_index['T4']] - data[:,feature_to_index['O2']]
```

1. SPECTROGRAMS-KAGGLE

- 128*256*4

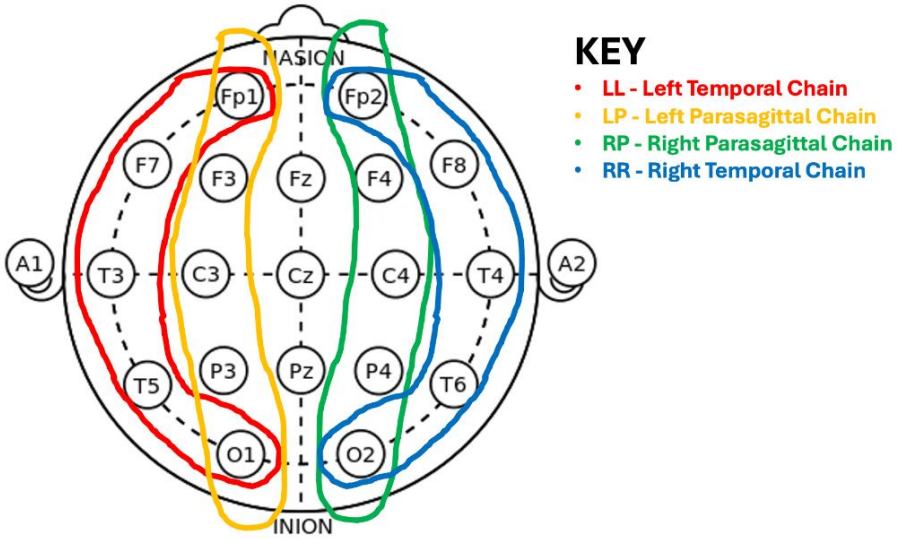
```
if self.mode=='test':
    r = 0
else:
    r = int(row['spectrogram_label_offset_seconds'] // 2)

for region in range(4):
    img = self.spectrograms[row.spectrogram_id][r:r+300, region*100:(region+1)*100].T

    # Log transform spectogram
    img = np.clip(img, np.exp(-4), np.exp(8))
    img = np.log(img)

    # Standarize per image
    ep = 1e-6
    mu = np.nanmean(img.flatten())
    std = np.nanstd(img.flatten())
    img = (img-mu)/(std+ep)
    img = np.nan_to_num(img, nan=0.0)
    X[14:-14, :, region] = img[:, 22:-22] / 2.0
```

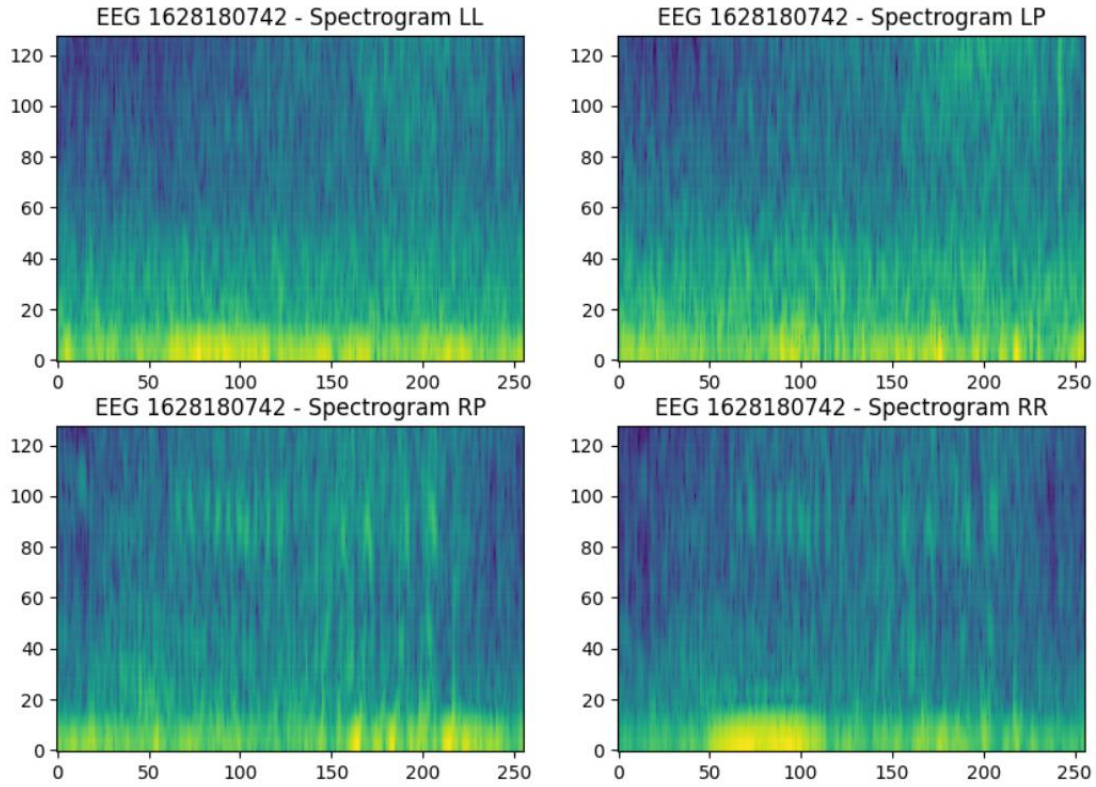


2. SPECTROGRAMS-EEG

- 128*256*4

```
LL Spec = ( spec(Fp1 - F7) + spec(F7 - T3) + spec(T3 - T5) + spec(T5 - O1) )/4
LP Spec = ( spec(Fp1 - F3) + spec(F3 - C3) + spec(C3 - P3) + spec(P3 - O1) )/4
RP Spec = ( spec(Fp2 - F4) + spec(F4 - C4) + spec(C4 - P4) + spec(P4 - O2) )/4
RL Spec = ( spec(Fp2 - F8) + spec(F8 - T4) + spec(T4 - T6) + spec(T6 - O2) )/4
```



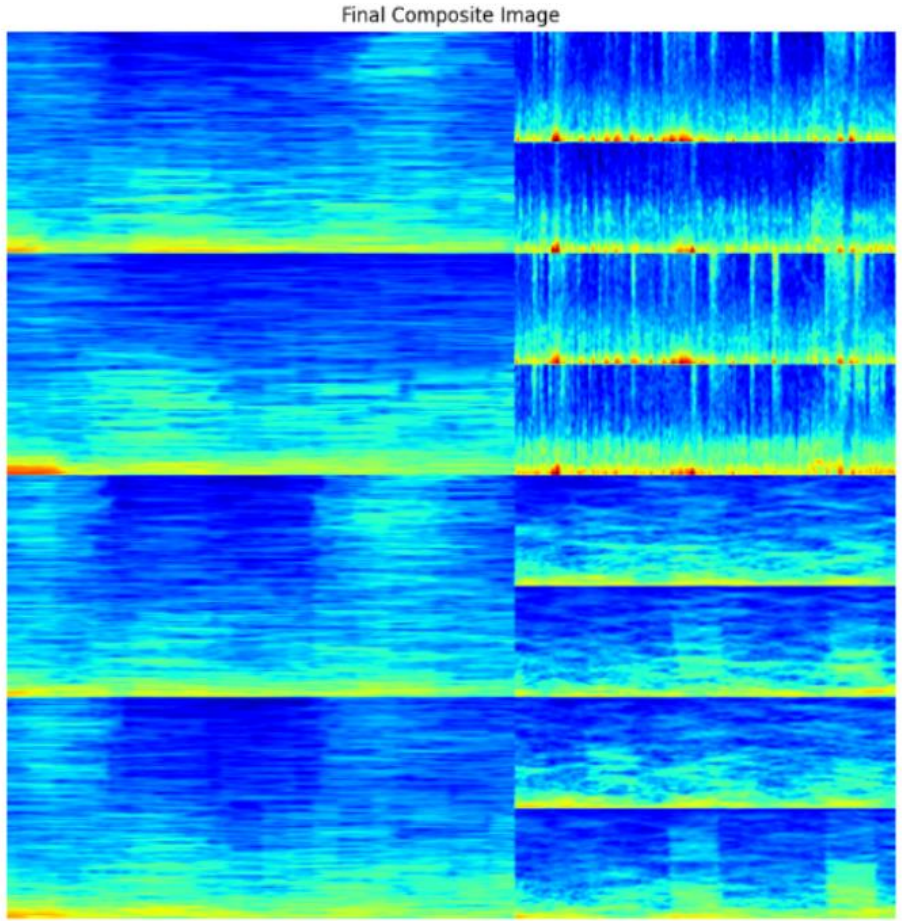


- https://www.kaggle.com/code/cdeotte/how-to-make-spectrogram-from-eeg

3. ZOOM SPECTROGRAMS

- 512*512*1

- Spec provided by the competition: 10 minutes of data.
- Custom Spec 1: 50 seconds of data (Zoom in the inference window).
- Custom Spec 2: 10 seconds of data (Even larger zoom in the inference window).

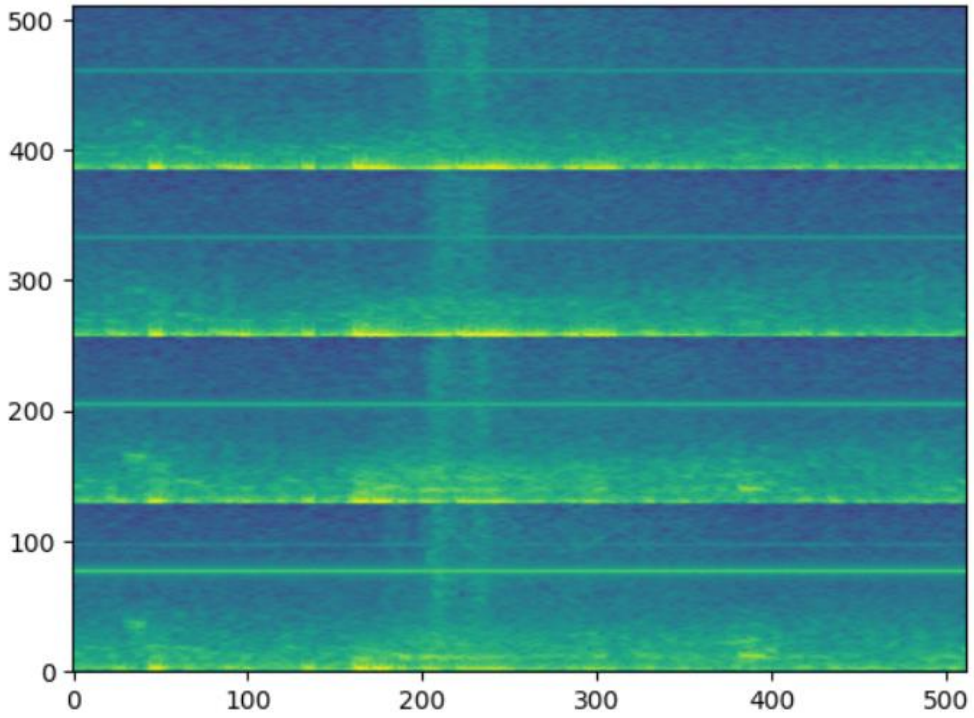


- https://www.kaggle.com/code/rafaelzimmermann1/hms-spectrogram-creation-using-gpu

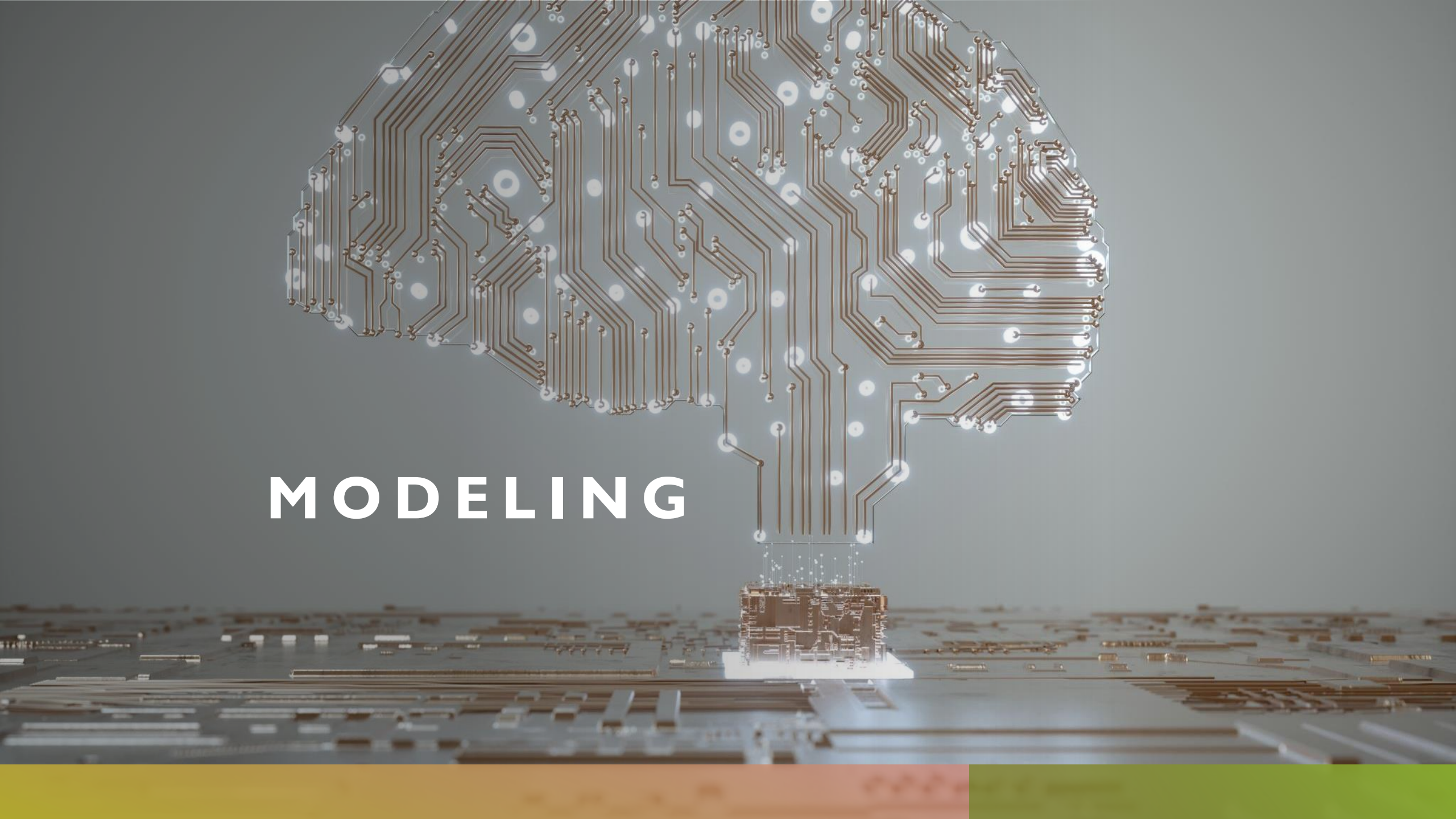
4. ANOTHER MAGIC SPECTROGRAMS

- 512*512*1

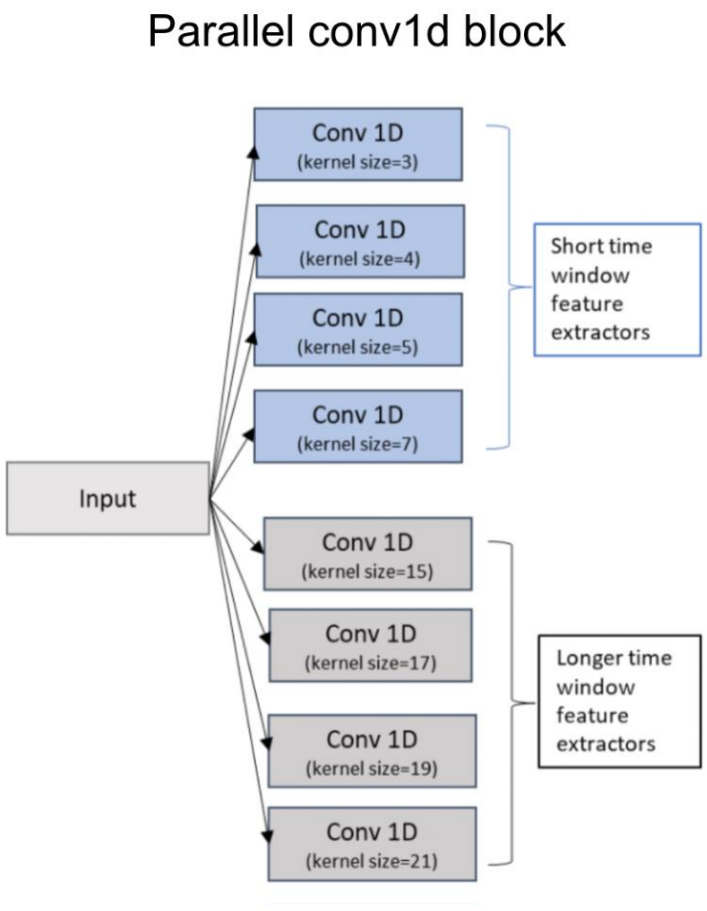
```
LL Spec = ( spec(Fp1 - F7) + spec(F7 - T3) + spec(T3 - T5) + spec(T5 - O1) )/4
LP Spec = ( spec(Fp1 - F3) + spec(F3 - C3) + spec(C3 - P3) + spec(P3 - O1) )/4
RP Spec = ( spec(Fp2 - F4) + spec(F4 - C4) + spec(C4 - P4) + spec(P4 - O2) )/4
RL Spec = ( spec(Fp2 - F8) + spec(F8 - T4) + spec(T4 - T6) + spec(T6 - O2) )/4
```

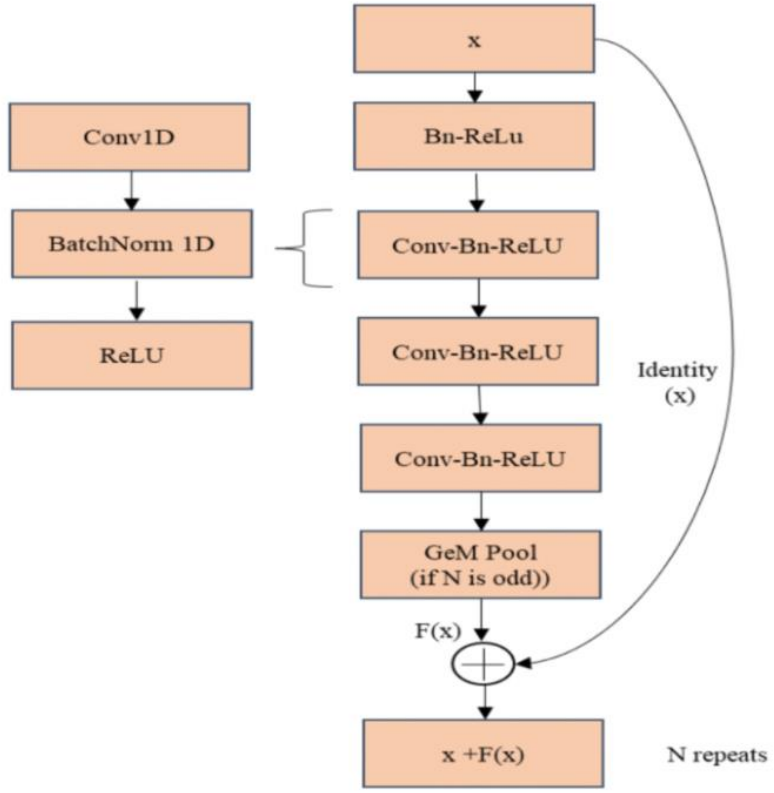


https://www.kaggle.com/code/naomideenen/train-spectrograms


MODELING

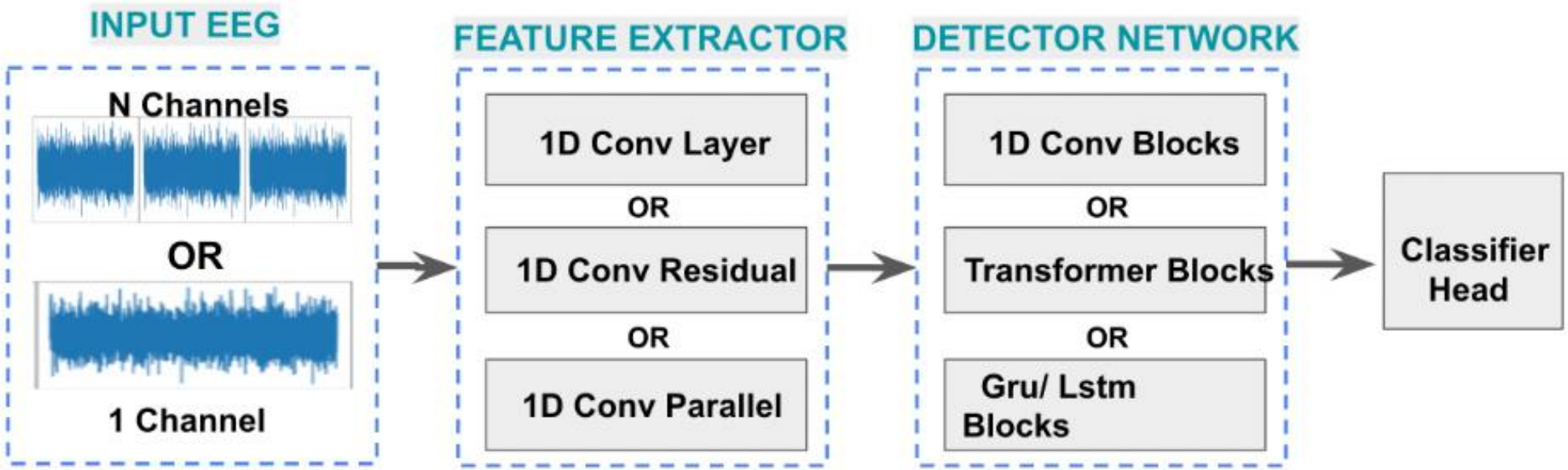
1D EEG MODEL


Parallel conv1d block
Input
Conv 1D
(kernel size=3)
Conv 1D
(kernel size=4)
Conv 1D
(kernel size=5)
Conv 1D
(kernel size=7)
Short time window feature extractors
Conv 1D
(kernel size=15)
Conv 1D
(kernel size=17)
Conv 1D
(kernel size=19)
Conv 1D
(kernel size=21)
Longer time window feature extractors


x
Conv1D
BatchNorm 1D
ReLU
Bn-ReLu
Conv-Bn-ReLU
Conv-Bn-ReLU
Conv-Bn-ReLU
GeM Pool
(if N is odd))
F(x)
x +F(x)
Identity
(x)
N repeats

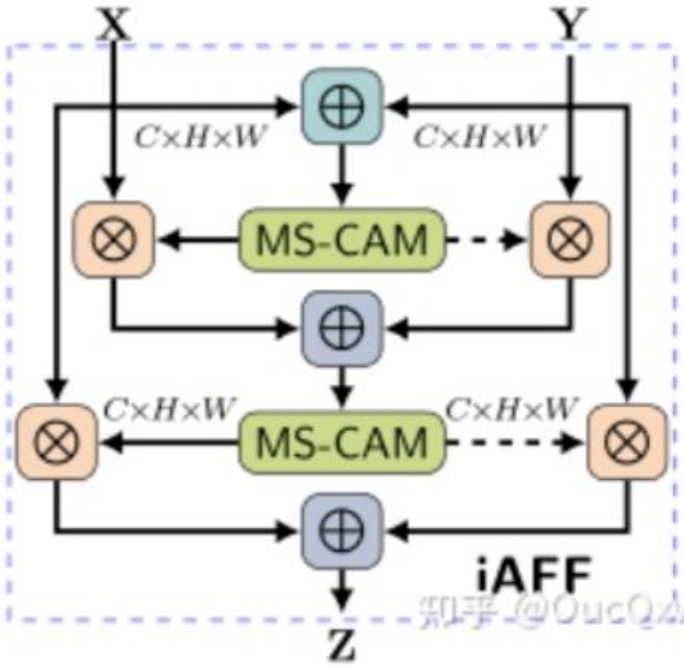
1D EEG MODEL

- Parallel Convolution Blocks + ResNet Like 1D Blocks + RNN head



MULTI-MODAL FEATURE FUSION IAAF

3.iterative Attentional Feature Fusion (iAFF)



- https://blog.csdn.net/amusi1994/article/details/120929046

TRAINING METHODS

GROUP K-FOLD

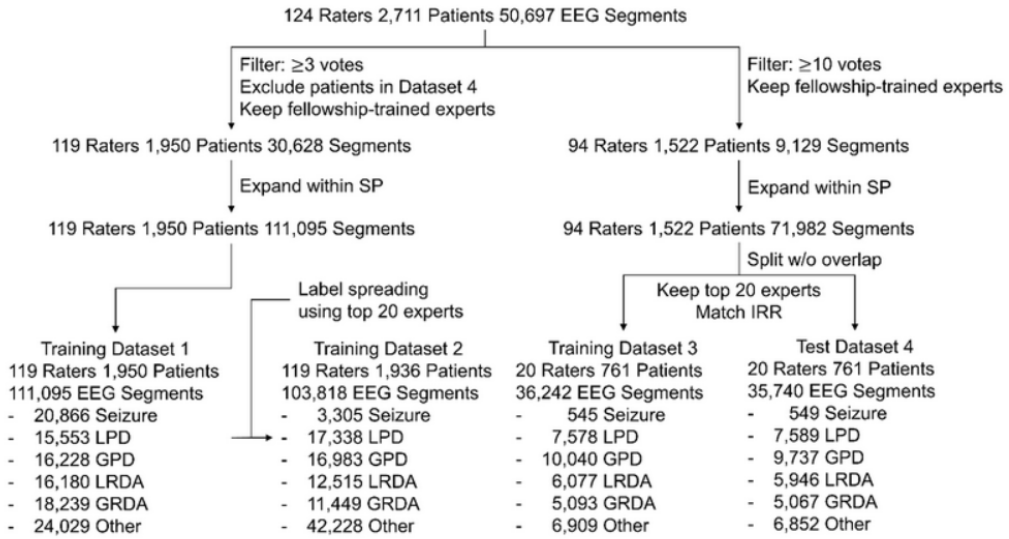
Typical scenarios for using GroupKFold include:

- Medical Research: When samples come from different patients, using GroupKFold ensures that data from the same patient doesn't appear simultaneously in both training and test sets, helping to evaluate the model's generalization ability on unseen patient data.
- Time Series Data: If data comes from specific time periods or events, GroupKFold can be used to ensure that data from the same time period isn't split between training and test sets, avoiding time leakage.
- Recommendation Systems: When dealing with user-item interaction data in recommendation systems, GroupKFold can be used to ensure that data from the same user doesn't appear simultaneously in both training and test sets, helping to evaluate the model's recommendation ability for new users.

```
gkf = GroupKFold(n_splits=config.FOLDS)
for fold, (train_index, valid_index) in enumerate(gkf.split(train_df, train_df.target, train_df.patient_id)):
    train_df.loc[valid_index, "fold"] = int(fold)
```

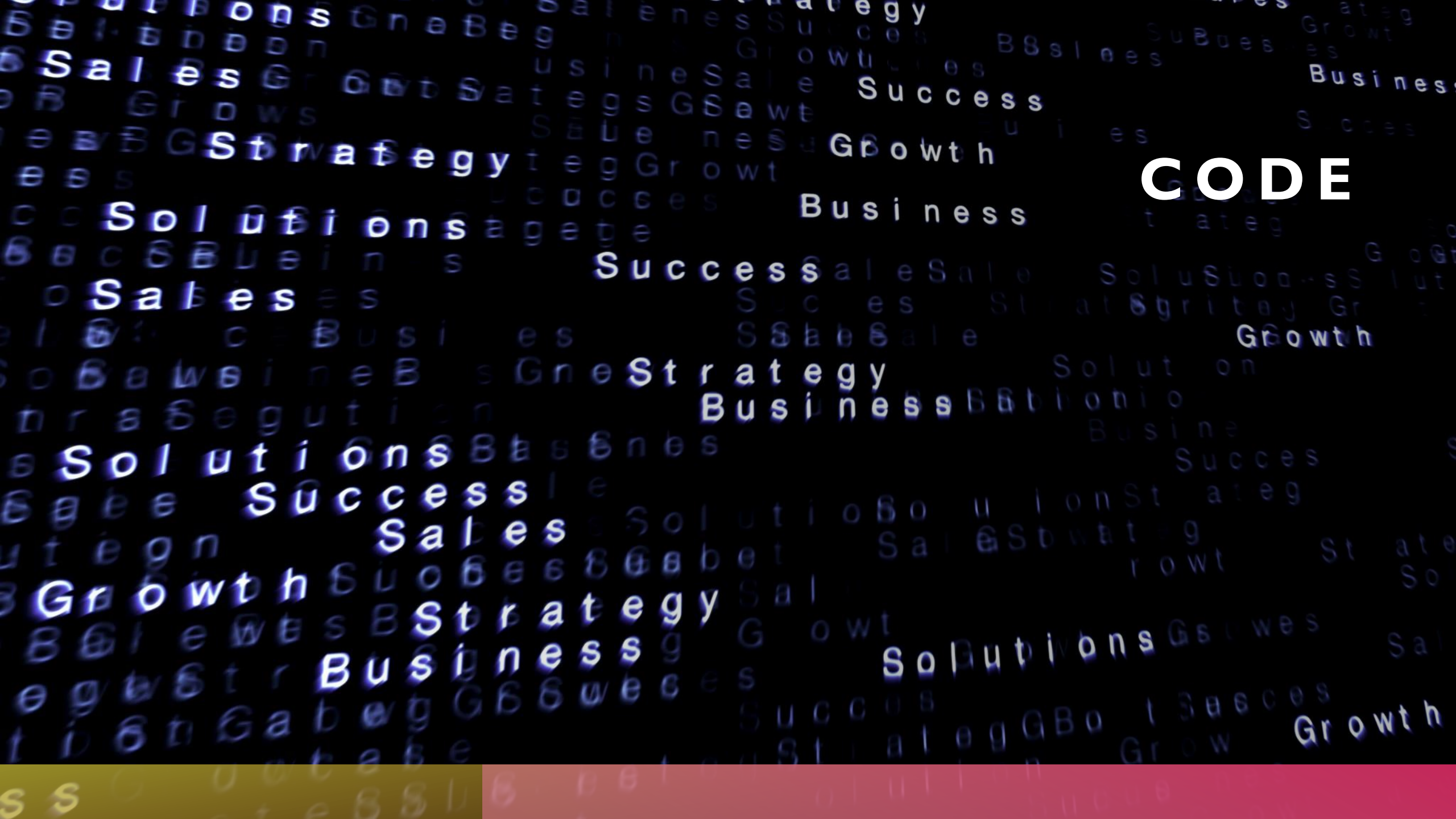
2 STAGE TRAINING

Figure S2. Flow diagram for creation of training and test datasets.




```
# ======== SPLIT ==========
if two_stage:
    train_folds = df[(df['fold'] != fold) & (df['total_evaluators'] >= 10)].reset_index(drop=True)
else:
    train_folds = df[(df['fold'] != fold)].reset_index(drop=True)

valid_folds = df[(df['fold'] == fold)].reset_index(drop=True)
```

- https://pubmed.ncbi.nlm.nih.gov/36878708/


CODE
Sales
Strategy
Solutions
Success
Growth
Business

IMPORTANT FINDINGS

- Version Management
- Data Understanding
- Model Improvement