Before the deep learning approach:

1. We visualized the data before the preprocessing, using a line graph, to understand the form of the data. For graphical simplicity the signals on the figure represent the mean value of all the sensors in each specific timepoint. (one image goes here)

2. Access Initial preprocessing: The data were uploaded on a drive folder and were accessed from colab python notebooks, where the scripts were written. After reading each individual folder of the training data, they were stored into a 32 X 248 X 35624 matrix, that translates to 32 files with 248 sensors in each file and 35624 timepoints in each sensor. The data were scaled using z-score normalization (citations 1,2 here) for each sensor individually. That included computing the mean (μ) and standard deviation (σ) of all the timepoints of each sensor and then subtracting the mean and dividing by the standard deviation. This way, the timepoints of each sensor have a mean of 0 and a standard deviation of 1 (citation 2). (mathematical equation of z-score normalization here).

Due to the large size of the data, downsampling was necessary to reduce computational load without reducing the quality of the data. In image and signal processing, decreasing the sampling rate is likely to create aliasing (citation 5). Nyquist Sampling Theorem (citation 6) states that in order to accurately capture a signal, you must sample at a rate that is at least twice the highest frequency in the signal (the Nyquist rate). This is to avoid an issue known as aliasing.

Aliasing occurs when high-frequency components of a signal are sampled below their Nyquist rate. These high-frequency components then appear as lower-frequency components in the sampled signal, which can distort the signal and lead to incorrect analysis or interpretation.

To prevent aliasing when reducing the sampling rate (i.e. downsampling), we applied a lowpass filter beforehand (citation 7). This filter removes the high-frequency components that could alias into lower frequencies during the downsampling process. By filtering out frequencies higher than half of the new sampling rate, we ensured that the Nyquist criterion is met for the downsampled signal.

After the filtering, we downsampled the data by 1/4, reducing the sample rate from 2034hz to 508hz. This way the computational load was reduced but the characteristics of the signal remained.

1.<https://arxiv.org/abs/2210.01081>

2.<https://www.researchgate.net/publication/336618030_Feature_Normalization_Effect_in_Emotion_Classification_based_on_EEG_Signals>

5.<https://arxiv.org/abs/2109.11839>

6.<https://www.semanticscholar.org/paper/Certain-Topics-in-Telegraph-Transmission-Theory-Nyquist/db0172576316dc748aea82e8f13fb4719ac933d5>

7.<https://ieeexplore.ieee.org/abstract/document/5999717>

3. We calculated some statistics per file and per sensor. For example, we measured the mean value of all the sensors for each timepoint and plotted a graph to see if there is a clear distinction between the lines of each class. We found out that the “rest” and “motor” classes are a little more distinguishable (compared to all others) which have the highest and the lowest values respectively, whereas “story” and “working memory” classes are not distinguishable from one another. (image goes here)

This minor distinction disappears of coure when we apply scaling such as z-score, since it standardizes the values and in the plot the lines are the one on top of the other.

We found out that min-max scaling maximizes the graphical distinguishability of the classes. (image goes here)

Before starting with the deep learning approach we decided to employ a machine learning algorithm to have as a baseline. For this, we used an SVM model that receives the mean value of all 248 sensors for each timepoint, after it has been scaled using min-max scaling, so the disparity of the classes is increased. The model was trained using hyperparameter tuning, grid search and achieved an accuracy of 75% which was encouraging considering the simplified training data (because we took the mean of all the sensors for each timepoint). (confusion matrix goes here)

4. Starting with the deep learning approaches, we initially considered the raw values of the MEG data as features and tried to train various types of deep learning models on them.

Wavelet transformation.

Wavelet transform is a mathematical technique used to decompose a signal into components at various scales. Unlike Fourier transform, which analyzes the signal in terms of sinusoidal waves, wavelet transform uses wavelets—small waves that grow and decay within a finite period. This makes wavelet transform particularly effective for analyzing non-stationary signals (like brain signals) where the frequency characteristics change over time.

We started with a continuous wavelet transformation approach (citation here). CWT works by comparing the signal to a wavelet at different scales and translations. Essentially, we stretched and shifted the wavelet to align it with different parts of the signal, which allowed us to analyze how the signal's frequency content changes over time. We defined 15 scales to use in the transformation. These scales determine the resolution at which we analyze the frequency content of the signal. Each scale corresponds to a certain frequency band, with lower scales capturing higher frequencies and vice versa.

Prior to the wavelet transformation we applied the preprocessing steps (z-score scaling, lowpass filtering and downsampling by 4) and after the wavelet transformation the data had shape 32 X 248 X 15 X 8906. A Convolutional Neural Network with 2 2-D convolution layers, 2 max-pooling layers, a dropout layer with 50% dropout and a dense layer for classification with softmax activation. The model achieved an accuracy of 62.6%. This accuracy is less than the accuracy achieved by the SVM model.

5. Feature extraction

Another approach we used is the feature extraction, where instead of using the raw values or their wavelet transformations as features we used some statistical data. In particular (citation here) suggest an approach where the timepoint series is segmented into parts (61 in our case) and then 8 statistical features are derived from each of these parts. These statistical features are the mean, standard deviation, maximum and minimum of the original signal as well as the average absolute value of the first and second difference, the skewness and the kurtosis of the original signal. The final shape of the data after the preprocessing, the segmentation and the feature extraction was 32 X 248 X 61 X 8. A Feedforward Neural Network with three dense and two dropout layers and a Convolutional Neural Network with 2 1-D convolution layers (seen in figure) were both tested in this approach. The feedforward neural network was fed with flatten data (248\*61\*8) whereas for the CNN we flattened the last two dimensions and fed it with 248 X 488 data.

The Feedforward neural network showed notable fluctuations in its training accuracy and ended up scoring 50% in the testing set, whereas the convolutional neural network outperformed it with 62.5% accuracy (figure). Unfortunately after a lot of hyperparameter tuning we did not manage to increase the accuracy levels of either of these 2 models.

6.

As a final approach we experimented with a combination of previous approaches along with manipulating the shape of the data to create more features. In particular after the regular scaling, lowpass filtering and downsampling we defined 5 frequency bands (citation 1), delta: (0.5, 4), theta: (4, 8), alpha: (8, 12), beta: (12, 30), gamma: (30, 80) for each signal sequence and computed 5 statistical features (mean, standard deviation, skewness, kurtosis and energy) for each of these 5 bands. This resulted in 25 features for each of the 248 sensors of each observation, making the data 32 X 248 X 25. Since the features decreased by a lot after extracting the statistics from the raw data, we decided to treat each sensor as a separate instance with its own label based on the label of the observation. So our data were reshaped in 32\*248 X 25 which is 8680 X 25 and the labels were repeated accordingly, meaning that in these 8680 instances each batch of size 248 had the same label. After this preprocessing we built a Feedforward neural network (and not a CNN because there is no spatial relationship between the statistical data of each sensor). The network is composed of 2 dense layers, a dropout layer and a classification dense layer with softmax activation. The model was trained with 400 epochs showing an increasing route with minor fluctuations. For the testing part, the most frequent label for each batch of 248 predicted labels was taken as the label of the specific observation. The result was surprisingly positive with 87.5% accuracy.

1.<https://www.researchgate.net/publication/309421571_Controlling_a_Servo_Motor_Using_EEG_Signals_from_the_Primary_Motor_Cortex>

Extracting features:

<https://dergipark.org.tr/tr/download/article-file/833143>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# New notes