"EEG + AI competition" explanation file

By Dr Guang Ouyang, Assistant Professor, Faculty of Education, The University of Hong Kong
If you have any questions about the competition, you can write to me at ouyangg@hku.hk

1. Data

1.1. The data is from EEG (brain signals) recorded during a single participant's performing of handwriting imagery (imagining the processing of handwriting but not actually doing it). The content of imagined handwriting is the 26 alphabets.

1.2. The handwriting imagery task goes like this: the participant sat in front of a computer monitor and wore an EEG cap recording the EEG signals from the head. Every three seconds, an alphabet (e.g., "k") was shown on the screen for 200 ms and the participant was required to imagine the process of handwriting this alphabet.

1.3. The data is organized as a typical machine learning setting, i.e., data samples + labels. There are 7800 data samples (300 samples for each alphabet). Each data sample is a 2d matrix with size of 24*801 where 24 is the number of electrodes (channels) on the EEG cap and 801 is the number of time points (equally distributed between -200 ms to 3000 ms) after the presentation time of each alphabet. Thus, the entire data samples were put into a 3d array with size of 24*801*7800. The label information was put into a single vector with matched length (7800). There are 26 different values (1 to 26) in the label, corresponding to each alphabet.

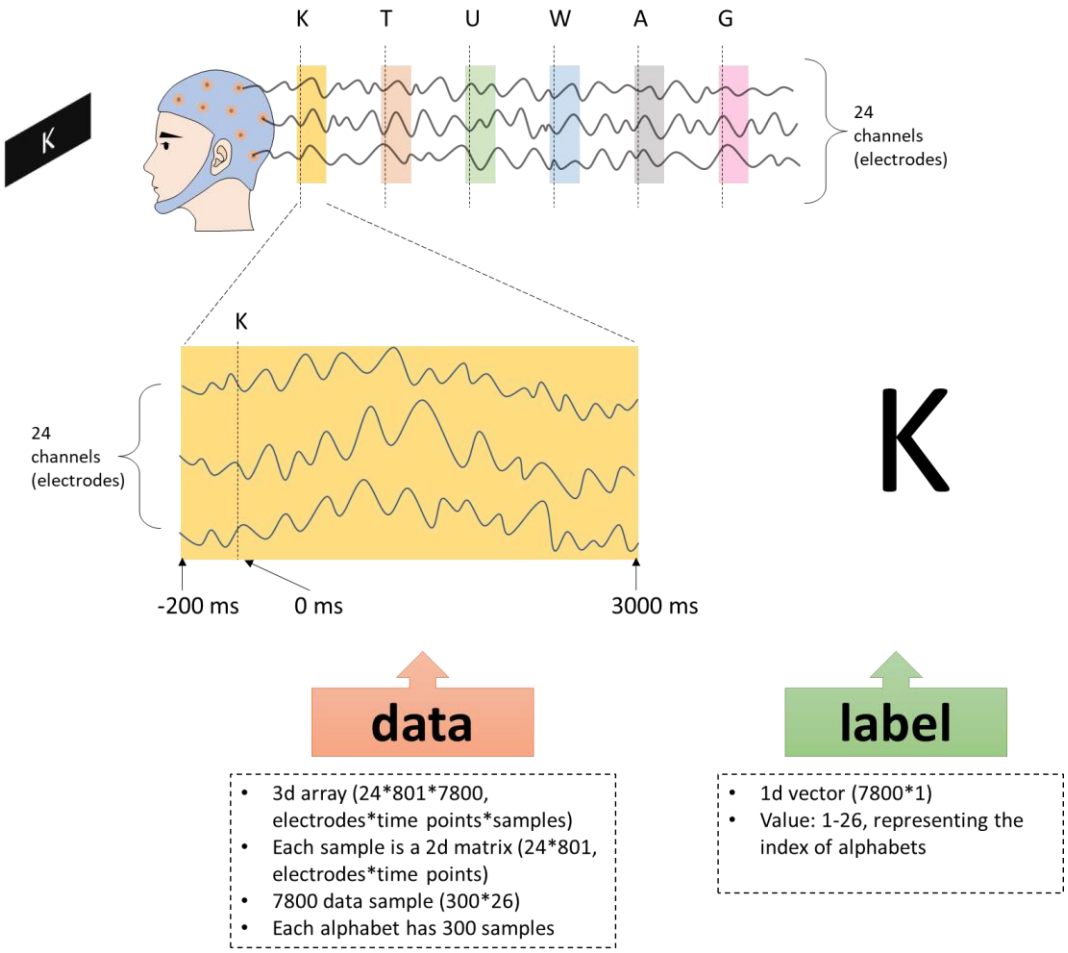


Figure 1.

1.4. The dataset has been pre-processed with major artifacts removed, baselined, and bandpass filtered between 0.1-45 Hz. Principal Analysis Method had been used to remove large-amplitude artifacts, so it is

possible that some data samples are not fully-ranked because a few PC has been removed (but relatively few).

1.5. The 24 channels (electrodes) with their labels rough positions are shown on Figure 2.

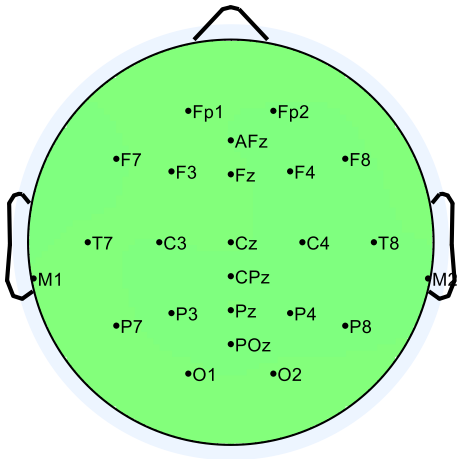


Figure 2.

1.6. The data can be downloaded from https://www.dropbox.com/s/2ug002c1btxkvvg/data_EEG_AI.mat?dl=0

1.7. The data is in Matlab format. If you load it into Matlab, you would see the following in the workspace.

Workspace

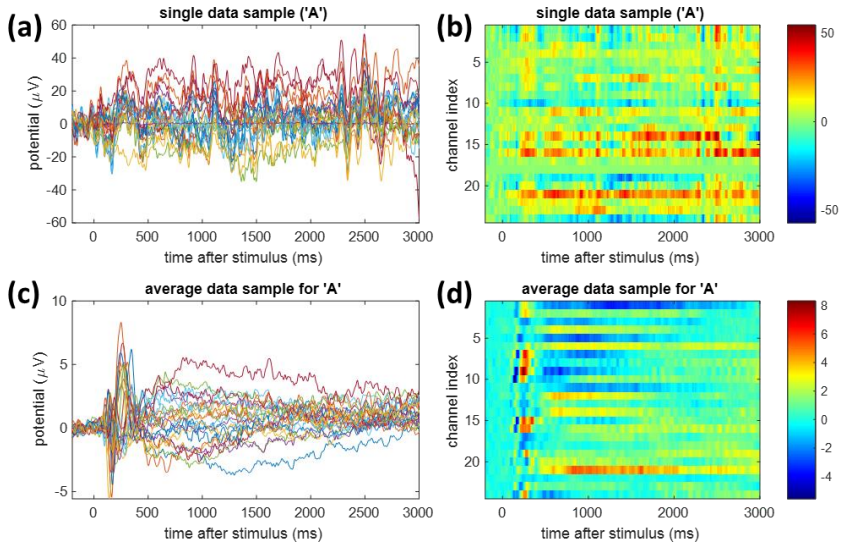
Name	Value
channel_labels	*24x1 cell*
data	*24x801x7800 double*
label	*7800x1 double*
time_points	*801x1 double*

1.8. The variable 'channel_labels' contains the labels of the channels (electrodes, see Figure 2). The variable of 'time_points' contains the time information for each single data point in the data sample ('data'). The meaning of 'data' and 'label' can be referred to from 1.3. These two are the key information for your classifier development.

1.9. You can also load the .mat file into different programming language environment (with mat importing package installed). If you load into Python (in Spyder) with the help of 'loadmat' package. You would see:

channel_labels	Array of object	(24, 1)	ndarray object of numpy module
data	Array of float64	(24, 801, 7800)	[[[8.18962261e+00 1.11856432e+01 -2.99568679e+00 ... -8.73955945e+00 ...
label	Array of uint8	(7800, 1)	[[1] [1]]
time_points	Array of float64	(801, 1)	[[-200.] [-198.5]

1.10. To get to know the data, we can try to plot the single trial data sample and the average pattern of data sample from the same alphabet. Figure 3(a,b) shows the data pattern for a single data sample (for the alphabet 'A') in two different ways. Figure 3(c,d) shows the data pattern for the average of all data sample for the alphabet 'A' (300 in total). It can be seen that the averaged pattern shows clear structure of brain response activity for processing and imagining writing the alphabet 'A'. However, the classifier should be developed to classify the data sample at single sample level (e.g., Figure 3a and 3b), which is what makes it challenging. The Matlab script for plotting Figure 3 is provided. You can play around with it (e.g., plotting different alphabets and comparing them). You can use the languages you are familiar with (e.g., python, R) to play around with the data.




```
figure;
plot(time_points,data(:,:,1)');
xlabel('time after stimulus (ms)');
ylabel('potential (\muV)');
title('single data sample (''A'')');

figure;
imagesc(time_points,1:24,data(:,:,1));
colormap(jet);colorbar;
xlabel('time after stimulus (ms)');
ylabel('channel index)');
title('single data sample (''A'')');

figure;
plot(time_points,mean(data(:,:,label==1),3)');
xlabel('time after stimulus (ms)');
ylabel('potential (\muV)');
title('average data sample for ''A''');

figure;
imagesc(time_points,1:24,mean(data(:,:,label==1),3));
colormap(jet);colorbar;
xlabel('time after stimulus (ms)');
ylabel('channel index)');
title('average data sample for ''A''');
```

Figure 3.

2. Competition rules

2.1. The task is to develop a classifier based on typical machine learning approach: training the classifier on a training data set and applying the classifier on a testing (or called validation) data set. This means that it is required to split the data samples (7800 of them) into training and testing set. You can also use k-fold method to do the training-and-validation work. In any case, you cannot apply your classifier to the data that has been used to train the classifier. The portion of testing data should be no less than 10%.

2.2. The three teams or individuals whose classifiers achieve the three highest accuracy (on testing data) will be awarded. Gold award (highest accuracy): 5000 HKD + a certificate; Sliver award (second highest accuracy): 2000 HKD + a certificate; Bronze award (third highest accuracy): 1000 HKD + a certificate.

2.3. You can participate individually or in a team. The maximum number of team member is three.

2.4. Eligibility: all undergraduate students in any university in Hong Kong.

2.5. Submit your code and full explanation of how to implement your code to me at ouyangg@hku.hk before September 30, 2022. If the description is not clear, we may arrange an interview with you.

3. Tips

3.1. The foundation for the possibility to develop a classifier that is able to accurately recognize the associated labels (alphabets) of EEG data samples is that the EEG samples have alphabet-dependent specificity. In other words, data samples that are from the same label (alphabet) would be more similar to each other than data samples from different labels. We can try to develop some similarity algorithm to confirm the alphabet-dependent specificity and to explore features that drive the specificity. Pearson correlation is a simple example of algorithm for characterizing similarity. **Figure 4** shows the correlation between the data patterns averaged from half of the same-label samples (150). The correlation value was calculated for each channel from the time window of [200-1000ms] and then averaged. This correlation matrix proves the existence of alphabet-dependent specificity. You can develop your own algorithm of similarity and use your algorithm to explore what kind of data features are driving the alphabet-dependent specificity, which may inspire your development of the classifier. But this step is not a necessary step.

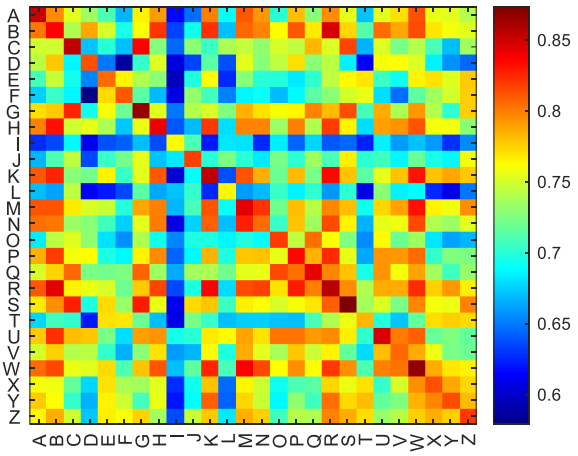


Figure 4

3.2. Since the data sample is a 2-D matrix which is in the same format of a gray-scale image, you may consider developing CNN-based classifier to learn the data feature and classify them, which may have higher chance of generating high classification accuracy. However, you are also encouraged the try whatever advanced classifier algorithms you think would be more powerful.

3.3. There are many (technically, infinite number of) different features you can extract from the data. With that said, you don't need to develop your classifier based on the raw data. You can perform any kind of transformation of filtering for secondary feature extraction before feeding the data features into your classifier in the training and testing process. Example of different features: different frequency bands, time frequency features, weighting different channels (electrodes), selecting different time windows, etc.

3.4. How high accuracy would a classifier achieve from the current data is unknown (it may never be known). However, you do know need to be frustrated if your classifier is not achieving a high classification accuracy. Remember, the chance level for the present data set is 1/26 = 3.8%. That means an arbitrary classifier would achieve this level of accuracy. So, if you can achieve a 10% accuracy, you should be happy about that. It is possible that a "not-so-high" accuracy gets an award. Of course, it would be fantastic if an amazingly high accuracy is achieved.

Readings that might be interesting for you:

Willett, F. R., Avansino, D. T., Hochberg, L. R., Henderson, J. M., & Shenoy, K. V. (2021). High-performance brain-to-text communication via handwriting. *Nature*, *593*(7858), 249-254.