Cognitive Science and AI

Assignment 2: Brain encoding and decoding with Nilearn

01/02/2022

CS9.432.S22

Instructions for submission

Maximum marks: 40

Deadline for submission on 8th February 2022 Before 12midnight

- You need to submit a notebook specified by Roll Number roll_no.ipynb in Moodle before deadline.
- Try to understand every piece of code and document the lines of code properly.
- Answers to each question could be listed out in separate document that should be sent along with the notebook. Please indicate roll number in the document.
- Your grade will depend on the correctness of answers and output. In addition, due consideration will be given to the clarity and details of your answers and the legibility and structure of your code.
- Include the assignment number, your name and roll number in the notebook as well for better identity.
- Don't wait until deadline.
- Make sure the assignment that you submit is your own work and understanding of the problem. Do not report/write ideas of your friends/colleagues.
- Do not copy or plagiarise, if you're caught for plagiarism or copying, penalties are much higher (including an F grade in the course) than simply omitting that question.
- Unless specifically permitted, collaborations are not allowed.

Objective

In this assignment, we replicate the results from James V. Haxby et al. (2001). "Distributed and Overlapping Representations of Faces and Objects in Ventral Temporal Cortex". In: *Science* 293.5539, pp. 2425–2430.

The visual object recognition experiment is conducted on the humans to understand the brain representations underlying ventral object visual pathway (see Figure 1), primarily, the ventral temporal cortex. We will replicate the idea of distributed and overlapping patterns of brain responses in humans while viewing objects like faces, cats, five categories of man-made objects, and nonsense pictures.

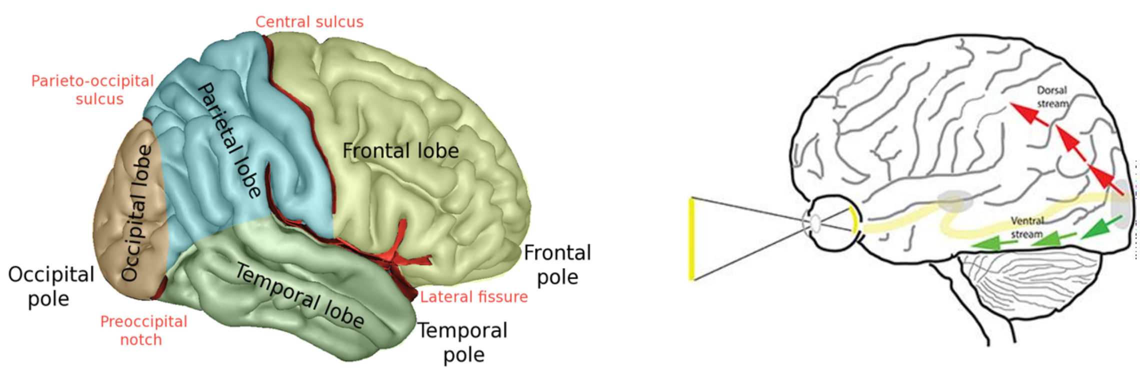

Figure 1: The lobes of the brain (left), the visual stream is segregated into ventral and dorsal streams from primary visual cortex V1 (occipital pole) (right). We are interested in visual representations of ventral stream area in temporal lobe like ventral temporal cortex. Figure adapted from (Sheth and Young, 2016). The source of the lobes of the brain `https://en.wikipedia.org/wiki/Lobes_of_the_brain`

The focus of this assignment is to get you familiar with standard fMRI analysis using General Linear Model (GLM) approach (K. J. Friston et al., 1994), in other words mapping brain responses to stimuli. The output images from GLM will form the basis of Brain decoding. Before getting into the assignment details, let's have a look at those concepts briefly: brain encoding and decoding. More in depth details could be referred to references wherever necessary.

Mapping brain response: Encoding

Standard analysis in task fMRI relates psychological manipulations to brain activity separately for each voxel also known as mass univariate analysis. It models the BOLD signal as a linear combination of experimental conditions – the General Linear Model (GLM, (K. J. Friston et al., 1994)) that produces the brain responses in reply to the experimental conditions, Figure 3. The BOLD signal forms a matrix

$$Y = X\beta + \epsilon$$

where Y is the acquired fMRI data of shape $\mathbb{R}^{n\times p}$, where p is the number of voxels and n is the number of scans/timepoints; X is the design matrix formed by k temporal regressors of interest $X \in \mathbb{R}^{n\times k}$. Each regressor is an indicator of occurrence of stimuli in the experimental design The design matrix also includes some nuisance confoundssuch as subject motion, scanner related as well as other noisy signals like low-frequency drifts present in the data. $\epsilon \in \mathbb{R}^{n\times p}$ denotes noise (K.J. Friston et al., 1998). The GLM is presented for one signal on Figure 2.

What we need to estimate are the $\beta \in \mathbb{R}^{k\times p}$ denotes the coefficients (the weight of the regressors), and $\epsilon \in \mathbb{R}^{n\times p}$ denotes a noise component. For each voxel j, this noise can be modeled as a Gaussian white noise with zero mean and variance σ_j^2. It is often modeled with some process (we will find out later) as the BOLD signal is auto-correlated in time domain. Then estimator becomes, $\hat{\beta} = X^{\dagger}Y$.

In our assignment, we use the Nilearn library (Abraham et al., 2014) to estimate such β maps or statistical maps.

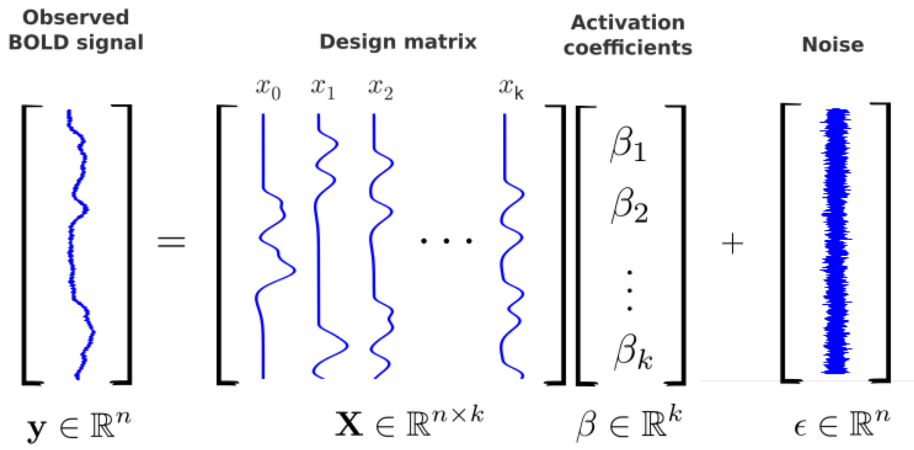

Figure 2: The GLM model for one voxel. The model expresses the acquired BOLD signal as a linear combination of regressors plus a noise term. Each regressor of the design matrix is the convolution of a reference HRF and the stimulus function. Each element of the (needed to be estimated) activation coefficients $\beta_1, \beta_2 ... \beta_k$ represent the relative amplitude of a given condition.

Decoding experimental conditions from statistical maps

Decoding predicts experimental conditions given statistical maps (Haynes and Rees, 2006). Unlike, brain encoding, it a multi-voxel pattern analysis that relates voxels responded to the psychological conditions (Figure 3). Such that this statistical classifier can be used to predict the mental state of a given individual after trained on large corpus of mental conditions on other individuals (Poldrack, Halchenko, and Hanson, 2009). With decoding, we can also bring new insights about how the mental processes are organized in the brain which is currently a big question in Neuroscience (Karl Friston, 2003).

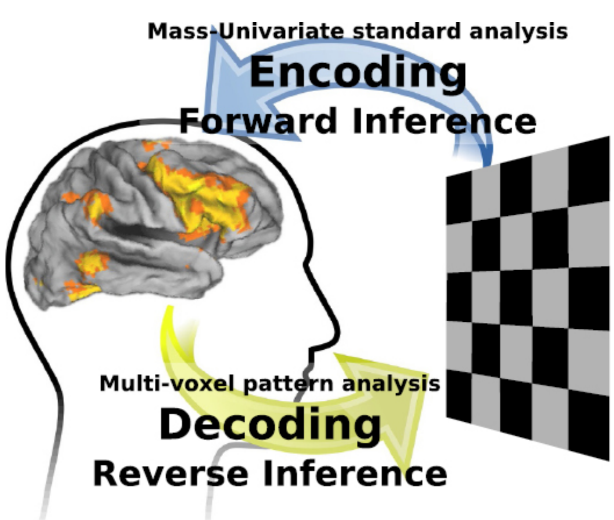

Figure 3: Encoding and decoding are the concepts that comes under fMRI data analysis. Figure adapted from (Varoquaux and Thirion, 2014).

Implementation of encoding & decoding

For all the implementations, we will exercise with Nilearn `https://nilearn.github.io/`. Generally, decoding based on fMRI data starts from unthresholded β maps as an input to classifiers. Such β maps needs to be estimated from fMRI data if not provided. In this assignment, we will learn how to estimate β maps.

Implementation starts with pre-processed task based fMRI data.

General Linear Modeling

- Download Haxby fMRI dataset with Nilearn. It includes files such as functional images and experimental sessions in txt format that are required as inputs for GLM. By default, only one subject data is returned. Let's first try GLM on single subject.

  Q1. What is the repetition time (TR) in seconds? (Hint: Have a look into this paper (Haxby et al., 2001)) Keep track of this value in seconds as it is important for GLM.

  ——————-

- Understand fMRI data: Load the functional image using relevant function from Nilearn. It should represent like Nifti1Image object.

  Q2. How many brain volumes are acquired in time? Pick any voxel within the brain and plot a time-course of one voxel or few voxels using Python library matplotlib $\rightarrow$ plt.plot. Visualization should appear in the notebook.

  Q3. What is the voxel resolution/dimension of the functional image? Is it isotropic or anisotropic?

  ——————-

- Load the session/experimental conditions file represented as ”session_target” using Pandas.

  Q4. List out the categories that are used in the experiments for visual object recognition task? Plot one image from any of the two categories.

  Q5. For how many runs/sessions are the experiments repeated? The information is provided with column name ”chunks” after loading the .txt file attributed to ”session_target”.

——————-

- While downloading the Haxby dataset, we also have various masks provided by the original authors. These masks are named as mask_vt.nii.gz, mask_face.nii.gz, mask_house.nii.gz and refers to ”ventral temporal”, ”face”, ”house” that are generated using GLM contrast based localizer maps. Use these masks as ROIs to keep the brain voxels that are responded to the stimuli and mask out the rest.

  From assignment 1, we have seen how to do masking using NiftiMasker provided mask as input. Plot mean time courses over these three masks using matplotlib → plt.plot. Analyze the BOLD waveform across three masks. Use these filtering choices in NiftiMasker as parameters for better interpretation of these signals. high pass=0.008 in secs, standardize=True, detrend=True, t_r=?, smoothing_fwhm=6. t_r should be a value that you found from Q1.

  Q6. After plotting the average signals from each of those masks (each 1D):

  – Does the signals appear like a block design or event related design?

  – From the raw timeseries signals, can you see which stimulus evoked a larger response? Is it vt or face or house?

Note: You can also apply zoom on these signals to help answer your questions.

Q7. What are the importance of high pass, standardize, detrend such parameters in fMRI time series analysis? Can you state their role? What could be influencing the results if such parameters are not specified?

From now, we look at how to estimate brain response maps using GLM with Nilearn. For simplicity, we manually take single session fMRI image data and accordingly prepare an events file for GLM analysis

- Make a design matrix **X** for GLM analysis. Prepare it in the form of dictionary. We analyze the whole fMRI image by separating out according to each session, sample screen shot is shown below. Append the code in below snapshots into a script that was used to load functional image and session_target.

```
# events will take the form of a dictionary of Dataframes

conditions_session = conditions[sessions == 1]

n_scans = len(conditions_session)
frame_times = TR * np.arange(n_scans)

duration = TR * np.ones(n_scans)

events = pd.DataFrame(
    {'onset': frame_times,
     'trial_type': conditions_session,
     'duration': duration})
```

Figure 4: A block of code for preparing events (single session) in a dictionary. This is given as input to FirstLevelModel (an implementation of GLM model).

After loading the txt file with Pandas, assign the column ”labels” into conditions and ”chunks” into sessions to execute the steps in Figure 4.

Q8. Observe the difference in the data array shape of conditions and conditions_session? What does that mean? You can print the shapes of both arrays and tell us why they are different?

- Initialize the GLM model

```
from nilearn.glm.first_level import FirstLevelModel
glm = FirstLevelModel(t_r=TR,
                      mask_img=haxby_dataset.mask,
                      high_pass=.008,
                      smoothing_fwhm=4)
```

Figure 5: GLM initialization from Nilearn. Note that we did not yet apply this model on functional image.

Q9. Why is smoothing necessary? What type of smoothing is implemented in the code?

- This step computes statistical maps/responses per condition.

After initializing (see Figure 5), next step is to fit the model on the functional image to observe the responses towards experimental conditions. After fitting, there are several interesting things to visualize: design matrix, statistical maps per condition. Sometimes, people call with the name contrast maps as they are related to different contrasts/conditions in the stimuli.

```
z_maps = []
conditions_label = []
session_label = []

from nilearn.image import index_img

session = 1

fmri_session = index_img(func_filename, sessions == session)

# fit the glm
glm.fit(fmri_session, events=events)

# statistical maps: one per condition
conditions = ['face', 'scrambledpix', 'scissors', 'shoe',
              'bottle', 'cat', 'chair', 'house']

for condition_ in conditions:
    z_maps.append(glm.compute_contrast(condition_))
    conditions_label.append(condition_)
    session_label.append(session)
```

Figure 6: GLM initialization from Nilearn. Note that we did not yet apply this model on functional image.

Q10. In the code, what is the role of nilearn.image.index_img? Why is it important to apply on functional image?

Q11. Dig into the FirstLevelModel and helps us understand what type of regression technique is used to estimate β maps? Which software is Nilearn dependent on for that regression technique?

Plotting the output attributes that are saved by the model after glm fit i.e., Figure 6.

Q12. Plot the design matrices attribute using nilearn.plotting.plot_design_m
What does each column represents? Does this makes sense with GLM design that is shown on Figure 2? Briefly elaborate how it made sense to you?

Q13. What is now the scanning length of fMRI per session?

Q14. What is compute_contrast? What it does? Plot the compute_constrast outputs for all conditions using nilearn.plotting.plot_stat_ma
with bg_img as Haxby data anatomical image. What is this output type z_maps means?

- Now, make a complete script and run the script for all sessions. The script should give us 96 contrast maps (mapped brain responses to stimuli), conditions label for decoding and sessions label for leave one session out cross validation.

Decoding: Classifying conditions given brain responses

Use the unthresholded β maps that are estimated above with Nilearn implemented GLM as inputs.

For implementation of decoding model, we use the same mask as you used to estimate contrast maps (see above in the code). Implement a decoding model from Nilearn nilearn.decoding.Decoder, model selection – leave one session out cross-validation from Scikit-learn and report the mean classification accuracies with classification model of your choice. We don't need a Deep Learning for now but standard machine learning models like SVC or Ridge that are readily implemented in Decoder Python object.

ROI based decoding on raw fMRI timeseries signals

Above, we did decoding based on whole brain mask. Now, in this segment, we try Region of Interests (ROIs) based decoding. In order to do that, we take the masks "mask_vt", "mask_face", "mask_house" and use each masks to restrict decoding to each of the ROI. Here, we don't need to estimate β maps but start with raw fMRI timeseries signals restricted to these ROIs.

See the example `https://nilearn.github.io/auto_examples/02_decoding/plot_haxby_full_analysis.html`. It tries to replicate (Haxby et al., 2001) paper. Have a careful read of this paper.

Q15. Tell us what this example tried to replicate in the paper. Does the classification accuracies per ROI across different categories makes sense with the outcomes of the paper? Convince us by writing a brief paragraph to assess your understanding.

References

Abraham, Alexandre et al. (2014). "Machine learning for neuroimaging with scikit-learn". In: *Frontiers in Neuroinformatics* 8.

Friston, K. J. et al. (1994). "Statistical parametric maps in functional imaging: A general linear approach". In: *Human Brain Mapping* 2.4, pp. 189–210.

Friston, K.J. et al. (1998). "Event-Related fMRI: Characterizing Differential Responses". In: *NeuroImage* 7.1, pp. 30–40.

Friston, Karl (2003). "Learning and inference in the brain". In: *Neural Networks* 16.9, pp. 1325–1352.

Haxby, James V. et al. (2001). "Distributed and Overlapping Representations of Faces and Objects in Ventral Temporal Cortex". In: *Science* 293.5539, pp. 2425–2430.

Haynes, John-Dylan and Geraint Rees (2006). "Decoding mental states from brain activity in humans". In: *Nat. Rev. Neurosci.* 7, p. 523.

Poldrack, Russell A, Yaroslav O Halchenko, and Stephen José Hanson (2009). "Decoding the large-scale structure of brain function by classifying mental states across individuals". In: *Psychological Science* 20, p. 1364.

Sheth, Bhavin R. and Ryan Young (2016). "Two Visual Pathways in Primates Based on Sampling of Space: Exploitation and Exploration of Visual Information". In: *Frontiers in Integrative Neuroscience* 10.

Varoquaux, Gael and Bertrand Thirion (2014). “How machine learning is shaping cognitive neuroimaging”. In: *GigaScience* 3.1.