Lab 3.2 - FMRI
Stat 214, Spring 2025

These instructions are specific to **Lab 3.2** and cover the background and deliverables needed to complete the assignment. Note that Lab 3 consists of three parts. Please also see the general lab instructions in **discussion/week1/lab-instructions.pdf** for information on how to set up your environment, write the lab report, and submit the final product.

Contents

1 Submission

Push to the`lab3` folder created in lab 3.1 to your `stat-214 GitHub` repository by **23:59 on Monday, April 28th**. Only submit **one** report per group. We will run a script that will pull from each of your GitHub repositories promptly at midnight so take care not to be late as late labs will not be accepted.

The 12-page limit is not in place for this lab. The page limit for this lab is 20 pages. The bibliography and academic honesty sections don't count towards this limit. This is an upper bound, your reports are not necessarily expected to be 20 pages.

Follow the general lab instructions in stat-214-gsi/discussion/week1/lab-instructions.pdf for more details. Please do not try to make your lab fit the requirements at the last minute! In your `lab3` folder, please provide everything (except the data) that is needed for someone else to be able to compile your report and receive the exact same pdf. Please follow the provided template for your report.

2 Academic honesty and teamwork

Academic honesty statement

We ask you to draft a personal academic integrity pledge, addressed to Bin, that you will include with all of your assignments throughout the semester. This should be a short statement, in your own words, that the work in this report is your own and that all sources you used are properly cited, including your classmates. Please answer the following question: Why is academic research honesty necessary? If you feel it is not, make a clear argument why not.

Collaboration policy

Within-group: In a file named `collaboration.txt` in your `report` directory, you must include a statement detailing the contributions of each student, which indicates who worked on what. After the labs are submitted, we will also ask you to privately review your group members' contributions.

With other people: You are welcome to discuss **ideas** with the course staff or other students, but your report must be written up and completed by your group alone. Do not share code or copy/paste any part of the writeup. If you discuss with other students, you must acknowledge these students in your lab report.

LLM usage policy

You are allowed to use LLMs (ChatGPT, GitHub Copilot, etc.) to *assist* in this lab, but are not allowed to use it for more than creating visualizations or helping correct grammar in the report. If we have reason to believe an LLM wrote a significant portion of your code (more than 5%) without your editing or iteration, or any section of your report word-for-word, this will constitute an honor code violation.

3 Lab 3.2 Overview

In Lab 3.1, we used bag-of-word and pre-trained embeddings to predict BOLD scores for voxels. In this lab, you will repeat the analysis from part 1 except that you will pre-train an encoder to generate your own embeddings. Then, you will fit a ridge regression model to predict voxels just as you did in 3.1

Why an Encoder? We focus on training an encoder rather than a decoder since the goal is to ultimately use these embeddings to predict voxels.

Useful links for pre-training encoders

1. `https://web.stanford.edu/~jurafsky/slp3/11.pdf`
2. `https://www.youtube.com/watch?v=kCc8FmEb1nY&t=1223s`

We have also provided some basic code for pre-training your encoder in the repository. Specifically, we provide the following three files:

1. `data.py` for loading and tokenizing your data. In this lab, we do not train our own tokenizer but use a pre-trained one. Specifically, we use `"bert-base-uncased"`. An example of loading and tokenizing your data is provided in `data.py`.
2. `encoder.py` consists of the general architecture for your encoder.
3. `train_encoder.py` Script to train the encoder that you need to implement.

Lab 3.2 Instructions

Data

Refer to instructions from lab 3.1 to get the data. The following are the instructions for the two main parts of the lab. Please carefully document your analysis pipeline, justify the choices you make, and place your work within the domain context. You should have an introduction and conclusion section in your report.

Part 1: Pre-training (80% of grade)

1. Implement the Encoder model in `encoder.py`
2. Implement masked-language model training for the Encoder model in `train_encoder.py`
3. Train the encoder, and plot training/validation loss curves. Do so for multiple hyper-parameters, and report the results for your tuning. What were the hyper-parameters you chose and why?

Part 2: Modeling & Evaluation (20% of grade)

We will now fit a linear model to each of the embeddings you generated in the previous step as you did in 3.1.

- Repeat the analysis you performed in 3.1 and fit a ridge model to predict across voxels.
- Does this model perform well across all voxels? What does this imply scientifically? What is a reasonable interpretation criterion for interpreting voxels according to PCS?
- Compare your results to embeddings you tried in lab 3.1. Perform a detailed analysis to compare the performance of embeddings. For example, do different embedding methods perform equally well across the voxels?

4 Peer Grading

So that you each have the opportunity to see alternative approaches to the labs, we will be doing peer-grading for this class.

You will each receive 2 reports from your peers to grade. A detailed rubric will be provided and you will be expected to provide both written feedback as well as a numeric grade on a variety of topics including communication, quality of data cleaning, relevance of visualizations, and reproducibility (can you easily re-compile their report).

After you have all submitted your own assignments (and shortly after the deadline), we will run a script that will automatically push two randomly selected reports into a folder called `lab3/peer_review/`. To retrieve your allocated reports, you will need to `git pull`. You will have one week to review these two reports and return your feedback in the form of a Google questionnaire that we will send by email to you. We will use these two grades for your report as a guide for grading, rather than as a final decision on your grade.