Description of Work

Thursday, May 17, 2018 1:49 AM

To go from Emory Box to MATLAB-accessible matrices of features, here is what I did

1. **Data segmentation**
    a. Every file was manually listened to for the times of each test for each subject. These times were entered into the spreadsheet 'Data Segmentation.xlsx'
        i. Notes: The spreadsheet is missing the 5 or 6 most recent subjects uploaded to Emory Box, but it still covers over 225 subjects. Many of these subjects are missing MCI/Control labels, unfortunately.
        ii. For the most part, the format of each subject's tests were the same, but there were a few exceptions that had to be manually corrected for out of order or unique tests.
        iii. In general, the people who recorded the data did a good job of not recording any voices, however: see next point.
        iv. There was a high variance of audio quality, even within subjects' files. In some files, you can hear the pen on the administrator's paper, and in others, the air conditioning, phone noises, or stopwatches can distort the subject's speech. Even in situations with no extra noise sources, the distance from the patient to the mic was perceptibly inconsistent, and the patients themselves, of course, had different voice levels.
        v. There are 4 sheets in the spreadsheet: one for the training set of subjects with extreme Moca scores, and three additional sheets, one for each study : Vascular, Calibrex, and Cedar.
    b. The MATLAB files 'dataTrain.m' and 'dataAll.m' use similar techniques to read in the 'Data Segmentation.xslx' spreadsheet and use that information to segment the audio files into smaller pieces and turn them into .wav files, which is important for feature extraction. These files are saved in the 'Voice Recording files TRAIN' for the training set and 'Voice Recording files ALL' for the whole dataset.
        i. Notes: At the beginning of each of the files is a list of parameters to be set to ensure that the processing goes as-intended. This includes the option to normalize the signals, the sheet names, the downsampling factor, the number of tests in each subject's file, and most importantly: the number of files to be processed in each sheet.
    c. The MATLAB file 'getFileData.m' extracts useful information for indexing and iteration in other programs, like the file names for both the entire data set and the training set, logical arrays specifying whether a given test was done by a given subject, the subjects' labellings as either MCI or Control, etc. All this information is packaged into 'getFileData.mat'.
2. **Feature extraction**
    a. With the dataset and training set segmented and files correctly following step one, feature extraction can be done. openSMILES was used for its out-of-the-box feature extraction capability for prosodic features and the GeMAPS featureset. Unfortunately, openSMILES was only accessible using the command line, since the gui always crashed. Instead, a series of windows batch files were written to automate the commands needed to do feature extraction on all the data. These csv file outputs were then stored in the 'Features' folder.
        i. Notes: the features are stored in csv files, where each csv file contains one feature set (GeMAPS or Prosodic) extracted for one test in either the whole dataset or the training set.
        ii. The top part of the csv file lists the feature names in-order, followed by a series of numbers whose rows represent an individual subject and whose columns represent individual features.
        iii. To redo the feature extraction, openSMILE must be installed on your machine and you must specify the path to the config files in the batch files before running them (~7

hours for the code to run).

b. Once the csv files were made available in the 'Features' folder, the final step was to read those files into MATLAB using the 'featuresProsodic.m' and 'featuresGeMAPS.m' scripts to create .mat files for further processing during analysis.
    i. Notes: The scripts create two data files: 'GeMAPS Features.mat' and 'Prosodic Features.mat'. Both contain a feature variable, which is a cell array listing the features extracted, in order, and two additional cell arrays. The cell arrays contain a list of tests in the first column and a corresponding array with features in the second column. One cell array is for the entire dataset, and the other is for the training set.
    ii. Right now, the biggest thing we are missing is a complete list of MCI/Control labels.

3. **Analysis**
    a. This is where I hand it off to you, Dr. Moore, unless you'd be okay with me helping you. However, I did write the file **'sampleSVM.m'** that demonstrates how the .mat files should be used and implements a simple SVM using the features extracted.
    b. Please, let me know what questions you have about methodology, code you don't understand, or anything else and I'll do the best I can to explain what I did.