CS 5310 Data Mining

Lab – Chapter 2 Managing and Understanding Data – Service Learning

Virtual reality mindfulness intervention for enhancing learning effectiveness research – service-learning

In this lab, we will process the data collected in a mindfulness intervention with virtual reality (VR) experiment and prepare the data for analyzing the effect of mindfulness intervention. This is a part of the continuous service-learning project supported by UHD Center for Community Engagement and Service Learning (CCESL). The procedure of the experiment is as follows.

**The procedure**:

*Learning procedure used in the experiment*:

The learning procedure is carried-out on computer. The subject logon to BrainGymmer.com account and choose a brain training game to play. The score of the game is recorded as the subject’s performance index.

*Pre-intervention activity*:

Each volunteer plays the BrainGymmer game as depicted above. The time taken in this step depends on how much time the volunteer uses to finish the game.

*Meditation*:

Each volunteer plays the VR TRIPP meditation game. The TRIPP demo is used to guide the meditation. The entire process take about 10 minutes.

*Post-intervention activities*:

Each volunteer plays the same BrainGymmer game as in the pre-intervention phase. The time taken in this step depends on how much time the volunteer uses to finish the game.

EEG data were recorded in all the above three phases using Compumedics Grael LT 34 channel recorder. Repeat the above procedure for each subject.

The EEG dataset we will process in this lab is saved in a Matlab file (with file extension .mat). The following is a code segment that suggests how to open the data file:

import os

import numpy as np

import pandas as pd

from scipy.io import loadmat

os.chdir(‘[The path to the following file]')

filename = 'Acquisition-15-data.mat'

data = loadmat(filename)

eeg = data['data']

“eeg” in the above code segment is a Numpy array with shape 34 X 63296. You need to transpose and save it in a data frame that has shape 63296 X 34 (63296 rows, 34 columns), where each column saves the EEG recordings of one channel.

Visualize the data (see instructions below).

We will use package “pyEEG” to compute power spectrum intensity (PSI) to be used in modeling. Download pyEEG package from <https://github.com/forrestbao/pyeeg> and follow the instructions on the website to install the package. The function we use in this lab is pin\_power(X, Band, Fs), where X is a list of 1-D real time series, Band is a list of boundary frequencies (in Hz) of bins. They can be unequal bins, e.g. [0.5,4,7,12,30] which are delta, theta, alpha and beta respectively. Set Band = [0.5,4,7,12,30]. Fs is an integer indicating the sampling rate in physical frequency, which is 1024 in our case. The function returns a list of PSIs of delta, theta, alpha and beta bands. We only use the alpha PSI (the 3rd value on the returned list).

For more information about pyEEG and its usage, visit:

<https://www.hindawi.com/journals/cin/2011/406391/>

<http://pyeeg.sourceforge.net/>

We will compute an alpha PSI for every second segment, which means X should be a list of 1024 EEG readings in each call to pin\_power(). In other words, one alpha PSI value is computed out of every second recording for one particular channel. If the entire EEG recording has 10 minutes, then there will be 10 X 60 = 600 alpha PSI values computed for one channel.

Repeat the above process to compute alpha PSI values for each of the 34 channels. Save the results in a data frame of 34 columns, corresponding to 34 channels.

Visualize the computed data (see instructions below).

Perform the following activities in Python:

1. Load data in input Matlab data file into a data frame in Python, as described above.
2. Compute the mean EEG values across 34 channels for each sample. This will generate an array of 63296 values.
3. Compute the standard deviation of the EEG values across 34 channels for each sample. This will generate an array of 63296 values.
4. Draw scatterplots of mean-stdev., mean, mean+stdev on the same plot, using different colors to distinguish different scatterplots. Here, “mean” denotes the means computed in Step 2, and “stdev” denotes the standard deviations computed in Step 3.
5. Compute alpha PSIs as depicted above.
6. Compute the mean alpha PSI values across 34 channels for each sample.
7. Compute the standard deviation of the alpha PSI values across 34 channels for each sample.
8. Draw scatterplots of mean-stdev., mean, mean+stdev on the same plot, using different colors to distinguish different scatterplots. Here, “mean” denotes the means computed in Step 6, and “stdev” denotes the standard deviations computed in Step 7.
9. Save the scatterplots created in Steps 4 and 8 in a Word document.
10. Save the alpha PSI dataset in a CSV file named “Acquisition-15-alphaPSI.csv”.

Turn in your Python source codes (the .py file), the output CSV file, and the Word document created in Step 9 in Blackboard by the due date. Also, log your service learning hours by following the web link in the “Service Learning” folder in Blackboard.