Homework # 3: In-Class Portion # 2

Due Date: Tuesday, October 8, 2019 by 11:59PM

Total Points: 25

This assignment provides practical programming experience with generating datasets for training neural networks. In particular, you will generate a dataset for training a neural network that estimates clean speech from noisy speech. **For the purposes of this assignment, please work individually**. However, feel free to ask basic questions to the instructor and classmates, but complete the assignment on your own. Feel free to search for information (inputs and outputs) about key Python functions, using the internet. From Canvas, go to Files and download all content from the HW#3_partII folder. You should also use code from the first portion of HW#3.

This assignment serves as one of the in-class components of HW# 3. Submit your answers to the questions and Python code to Canvas. **Be sure to include a README that explains each major script/function.** Answer all questions within the code. Feel free to create separate functions as needed. **Be sure to comment your code and to submit all files to Canvas**.

All assignments must be submitted on time to receive credit. No late work will be accepted, unless you have a prior arrangement with the instructor.

Question 1. [10 POINTS]

Dataset Generation: For this problem, you will generate the noisy and clean speech dataset that will eventually be used to train a deep neural network (DNN). Often times, noisy speech data is artificially created as this gives the researcher more control over the experiments. Follow the steps below, be sure to write Python code when necessary. You'll likely want to save these to a Box Folder or somewhere on the IU Carbonate Deep Learning nodes. **Submit your code along with screenshots of the file structure for your data.**

- Download the IEEE male and IEEE female speech signals from the Courses Box folder. Speech Data/IEEE. Divide the speech data into training, development and testing folders. Use the first 500 sentences from each of the male and female signals for training, use the next 100 sentences each for development, and the next 100 each for testing purposes. In Python, **resample the data (if necessary) to a sampling rate of 16 kHz.**

- Download the three noise files from the Course's Box folder: Noise Data. In Python, divide each noise signal in half, were the first half of each signal will be used to generate noisy speech for the training and development datasets, while the last half of each noise signal will be used for the testing dataset. **Resample the data (if necessary) to a sampling rate of 16 kHz.**

- For each speech signal from the training (and development) data set, randomly select (in Python) an equal length, contiguous segment from the noise training (and development) signal. For each speech and noise signal pair, use your function from HW#3 part I to generate noisy speech at signal-to-noise ratios (SNRs) of: -3, 0, and 3 dB. Meaning, for 1000 training signals (500 from male and 500 from female), 1 noise segment, 3 noise signals, and 3 SNRs per signal pair, there should be $1000 \times 1 \times 3 \times 3 = 9000$ training signals. **Be sure to keep track of the corresponding clean speech signal for each noisy speech signal. You may want to rename the files and output as wav files to a different directory.**

- Repeat this process for the development and testing data as well, but be sure to use the appropriate noise signal (e.g. training vs. testing).

Question 2. [15 POINTS]

Data Loading and Preprocessing. Follow the Data Pre-processing tutorial (e.g. datapreprocessing.ipnyb) to create data loading objects for the training, development and testing sets. Note that there are comments in the code that need to be addressed. In addition to this, perform the steps below:

1. For the training and development datasets, modify the code to compute the means and standard deviations across all time frames and for each frequency. Plot (in separate subplots) the resulting mean and standard deviations for both sets versus frequency (in Hz). Be sure to correctly label the plots.

2. Likewise, compute the minimum and maximum values for each frequency, across all time frames of the training and development datasets (e.g. one min and max vector for each set). Plot (in separate subplots) the resulting min and max vectors for both datasets versus frequency (in Hz). Be sure to correctly label the plots.

3. Using the mean and standard deviations, perform data standardization on the training and development datasets. After this is performed, recompute and plot the mean and standard deviation of the standardized data for both datasets.

4. Using the minimum and maximum vectors, perform data normalization on the training and development datasets. After this is performed, recompute and plot the minimum and maximum values of the normalized data for both datasets.