Audio Testing Spec

Danny Maclean

Purpose

This repository has purposes: Create an ABX structured test between 2 scenarios to give to listeners, analyse the test and runtime data to determine which scenario was better.

ABX Testing

ABX testing is a way of testing audio that involves three different audio files, an A file, B file and an X file. First, users listen to A and B and writes down which they prefer and by how much, on a scale from 1 to 5. Next, users listen to the X file, which is a copy of either A or B, and answers which it is a copy of and how easy it was to distinguish. Finally, there is a space to leave notes on the test case such as, why one was preferable, why one was less preferable and so on.

Creating ABX Tests

This script creates an ABX test for 2 scenarios. An example scenario that could be compared is Accession with max jitter buffer of 500ms under 200ms jitter versus Accession with max jitter buffer of 1000ms under 200ms jitter.

Creating Test User Flow

1. From the command line cd to the root directory and run manual\_testing.py script
2. A testcases directory and answer\_key.yml is then added to the root directory

Manual Test Command Line Arguments

* -o, output\_base\_path: (optional) Absolute file path to location to save test directory and answer key
  + Default: root directory
* -s1, scenario\_one: Absolute file path to location of first scenario.
* -s2, scenario\_two: Absolute file path to location of second scenario.

Scenario One and Scenario Two

Please note that manual\_test.py makes 3 assumptions about these file paths.

1. Both scenarios contain the same amount of wav files.
2. The wav files in both scenarios have a one to one correspondence between each other. Each test case contains a pair of files, one from each scenario. This pair is made by matching files between scenarios with the same names
3. There are not more than 25 wav file pairs. Anymore additional wav files will be ignored

Responses Format

The listener will be provided with an excel sheet to document their answers and notes about the test. The excel sheet is labelled “responses.xlsx” and contains 4 parts for the listener to answer for each test case. More detail is provided in the response sheet

* Preference: Which file was preferred: A or B? An answer must be given, even if it not sure
* Rating: How much better was the preferred recording on a scale from 1 to 5
* X: Which of A and B was X? An answer must be given
* Distinguishability: How easy was it to tell on a scale from 1 to 5
* Notes: Any thoughts or comments about the audio or test cases can be documented here. This is optional

Answer Key

After the test directories are created, an answer\_key.yml is created in the root directory. It is important to keep this file so the listener’s answers can be graded afterwards.

Answer key structure:

* Scenario One Latency: this is the latency for the first scenario files if input
  + The title is the name of the subdirectory passed in
  + Cross Correlation Coefficient: This is the maximum correlation coefficient indicating how related the two scenarios were at that time. A high positive value means they are almost identical, zero and negative values indicates a very low correlation
  + Time Latency: This indicates how long it took a receiving phone to hear the audio from the transmitting phone.
* Scenario Two Latency: this is the latency for the first scenario files if input
  + The title is the name of the subdirectory passed in
  + Cross Correlation Coefficient: This is the maximum correlation coefficient indicating how related the two scenarios were at that time. A high positive value means they are almost identical, zero and negative values indicates a very low correlation
  + Time Latency: This indicates how long it took a receiving phone to hear the audio from the transmitting phone.
* Question Number
  + A Value: this note which scenario, A belongs to
  + B Value: this note which scenario, B belongs to
  + X answer alpha: This is the alphabetic value (A or B) of X

Grading and Analysing User Flow

1. Confirm that the anwer\_key.yml and responses.xlsx are in the root directory
2. From the command line, cd to the root directory and run grade.py
3. The test will output a test\_results.py file

Grade.py Command Line Arguments

* -fp, file\_path: (optional) Absolute directory path to a collection of tests results from a variety of users. The program will crawl all files in this directory and grade files with 2 regular expression keywords: “responses” and .xlsx. Each graded file will have a corresponding “test\_result.yml” with the file name appended to the start of the filename. Please note that the script will grade all files in subdirectories with the keywords and additionally files with other characters if the string contains the keywords.
  + Default: The root directory will be searched.
* -o, output\_path: (optional) Absolute file path to location for test results documents.
  + Default: The root directory

Test Results Contents

* Scenario One: File Path
  + Cross Correlation Coefficient: This indicates the correlation between the input and output signal. Please refer to ATAT spec for further details
  + Time Latency: This is the average latency of signals during the call. Please refer to the ATAT spec for further details
* Scenario Two: File Path
  + Cross Correlation Coefficient: This indicates the correlation between the input and output signal. Please refer to ATAT spec for further details
  + Time Latency: This is the average latency of signals during the call. Please refer to the ATAT spec for further details
* Total Correct: the amount of correct question to total questions
* Total preference for Scenario One: number of correct questions that preferred Scenario One
* Total preference for Scenario Two: number of correct question that preferred Scenario Two
* Degree of preference for Scenario One: Average of preference values of all correct questions that preferred Scenario One
* Degree of preference for Scenario Two: Average of preference values of all correct questions that preferred Scenario Two
* Question: The number displayed is the question number
  + Correct: this is a Boolean indicating whether the user got the x value correct
  + Correct Answer: the correct answer and the scenario it belongs to
  + User Answer: the user’s answer and the scenario it belongs to

Grading Multiple Responses at Once

To grade a collection of tests at once, organize all user responses into one directory and provide the absolute file path to the directory as a command line argument to the script. The script will then grade any file with the key word “responses” in the name and file extension .xlsx. Note that the user’s tests can be organized into subdirectories, the script will crawl all files in the path and only grade ones that meet the two requirements. Every response document will have a separate output “test\_results.yml” file in its directory, with detailed results for every question, and a “master\_test\_results.yml” in the root directory of the script with only the test statistics. Note that the answer\_key.yml file should still be in the root directory.