School of Languages, Linguistics and Film Assessed Coursework Coversheet

For undergraduate (BA) modules coded:
CAT-, COM-, EAL-, FLM-, FRE-, GER-, HSP-, LAN-, LIN-, POR-, RUS-, SML-

Please read and note the following guidelines:

1. To assist with anonymous marking, please use your nine-digit student ID number only: do **NOT** use your name anywhere on your coursework.
2. Normally you will be required to submit one electronic copy of coursework via the module's QMplus area. Most deadlines in this School are set for a Sunday night (23:55). You will be informed by the module organiser of any exceptions to this procedure, either regarding the time or method of submission. It is your responsibility to ensure that you know and meet the submission requirements for each piece of coursework.
3. You must keep a copy of all coursework you have submitted.

4. Extensions to deadlines may ONLY be granted by the Senior Tutor for your department. In order to be granted an extension, you must submit a claim for Extenuating Circumstances BEFORE the coursework deadline. SLLF has an online EC claim form. Details and links to the form can be found on QMplus School of Languages, Linguistics and Film Landing Page.

5. Late submission, without an agreed extension due to extenuating circumstances, will be penalised according to the QMUL regulations relevant to your level of study.
6. Work submitted within 7 DAYS of the deadline will be accepted but subject to a late submission penalty against the marks awarded. The work will be marked normally, and then a late submission penalty of five marks (or 5% of the marks if not marked out of 100) per 24 hour period will then be applied.
7. Work that is more than 7 DAYS late will not be accepted and will not be marked and will receive a mark of ZERO.

You are reminded that plagiarism, that is copying someone else's words or ideas without attributing them to that person, is cheating. This is a serious examination offence and at the very least will result in a mark of zero being awarded for this piece of work; it could result in your expulsion from Queen Mary.

By handing in this coursework you acknowledge that it represents your own, unaided work and that you have appropriately acknowledged all sources.

Please complete the following details:

Student ID Number:(9-digit number): 190245566
Module CODE and TITLE: LIN6209. Coding For Linguists
Title of Coursework: Mini-project assignment
Essay no: 2
Number of words written:
Module Organiser: Peter McGinty
Seminar Tutor (if applicable):

Please continue your coursework on the next page

Measuring the acoustic features of Smiled Speech vs Neutral Speech

Initial idea

The initial proposal was to build a supervised machine learning classifier using data from previous research. This programme would take in an object and use a series of parameters such as acoustic measurement values (e.g. 300Hz) to classify this object as “smile” or “neutral”. This sort of measurement and classification has already been done. Based on research papers I had read, I expected to be able to obtain data that had been already formatted into a usable database. I asked the authors of several projects and corpora to share their data with me. I expected to get a spreadsheet with data in the following format or similar:

1) ‘vowel, duration, f0, f1, f2, f3, label’

After enquiring, the data that was available to me was in a completely different format. I was able to get a corpus of audio files and corresponding text files containing timestamps for each phoneme. Files where organised in directories but not labelled individually. Given that the data available was so different from my expectations, I had four options:

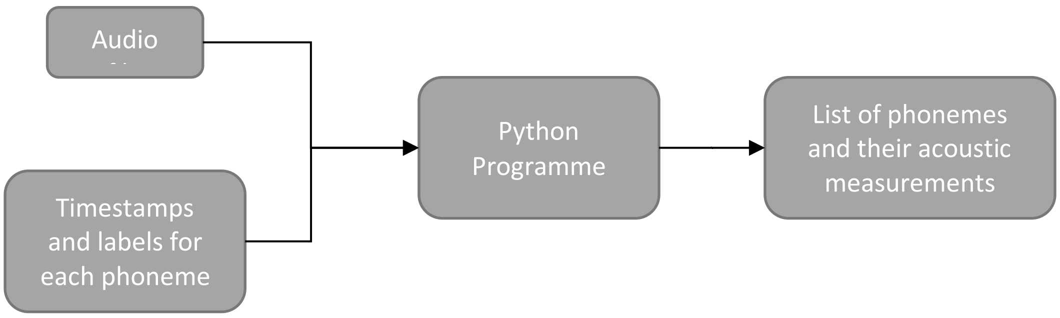
2) *New course of action*
   a. Do every measure myself manually for every vowel, build the dataset, then continue with the project as planned.
   b. Build a more complex model that works directly on audio files.
   c. Switch the main objective of this project from the classification to the data extraction and preparation process, and develop a programme that can create a database of objects like the one described in (1).
   d. Find a completely new project.

Options (a) and (b) were discarded because they are outside of the scope of this assignment; they would have required more time, resources, and skill than this project called for. I did not choose option (d) as this topic (i.e. the distinction between smiled and non-smiled speech) is very interesting to me. I decided to go with option (c) and focus on the extraction and preparation of data, as this is a useful skill that can be applied to other projects. The incremental and iterative nature of the project – solving many small problems before being able to tackle a big question – is also reflective of the reality of development projects.

New project scope

The goal of this software is to take acoustic measurements from audio files and present it in a format that can be used for multiple analyses. The acoustic measurements will be taken at times specified in a text file.

Figure 1. *High level description of the input/output structure of the programme*:


Audio
Timestamps and labels for each phoneme
Python Programme
List of phonemes and their acoustic measurements

How to find the data and code for this project

All the code and necessary data to run it have been uploaded to GitHub and can be found here. This repository contains:

- The following Jupyter Notebooks:
  - Session 2, the first coding session for this project.
  - Session 4, functions to iterate through directories.
  - Final Notebook, containing the final code.

- The following folders:
  - Smile, containing *wav* and *lab* folders with audio and text files respectively for smiled sentences. Each subfolder contains 20 files, as I did not have permission from the owners to share the full data from the corpus.
  - Neutral, same as *smile* but with neutral phrases.

Development diary

Session 1

This session was mainly dedicated to mapping out the structure of my programme. This was done mostly on paper.

As explained above, I was working with data collected during previous research on smiled speech. Dr. El Haddad from the University of Mons (Belgium) shared with me AmuS, a corpus of amused speech. The data was organised in folders in the following way:

3) *How data was shared with me*

My working directory

- Smile
    - Wav
        - Audio001.wav
        - Audio002.wav
        - Etc.
    - Lab
        - Text001.lab
        - Text002.lab
        - Etc.
- Neutral
    - Wav
        - Audio001.wav
        - Audio002.wav
        - Etc.
    - Lab
        - Text001.lab
        - Text002.lab
        - Etc.

There were four pairs of smile/neutral folders, one for each speaker. Each audio file contains a short sentence. The corresponding text file contains timestamps for the beginning and the end of each phoneme (sound). See (4) for an example of a text file:

4) *Extract from "SpkBeng_0001.lab"*

```
0 2400000 sil
2400000 3500000 sil
3500000 4900000 ao
4900000 6100000 th
6100000 7700000 er
7700000 8000000 ah
8000000 8300000 v
8300000 9100000 dh
9100000 9600000 ax
```

Given the amount of data I had available, I decided to reduce the scope of this project to only one of the speakers in the corpus. That is, one pair of smile/neutral folders. I broke down the project into three parts:

5) *Parts of this project*

- Work on ONE file: Extract information from one text file (.lab) into Python. Take acoustic measurements from corresponding audio file (.wav).
- Collect all filenames: Iterate through directories to collect a complete list of all files in text-audio pairs.
- Iterate through list of files taking measurements of each one and building a final database.

The ultimate goal was to develop a programme that could create and populate a database such as:

6) Goal database

Phoneme	Initial_time	End_time	Duration	F0	F1	F2	F3	Label	Audio_name
X									
Y									
z									

<u>Session 2</u>

In this session I started coding. I started a Jupyter Notebook and wrote at the top the provisional steps I thought I would follow (7). My main goal for this session was to accomplish the first part of the project (5): process a text file and take measurements from one audio file.

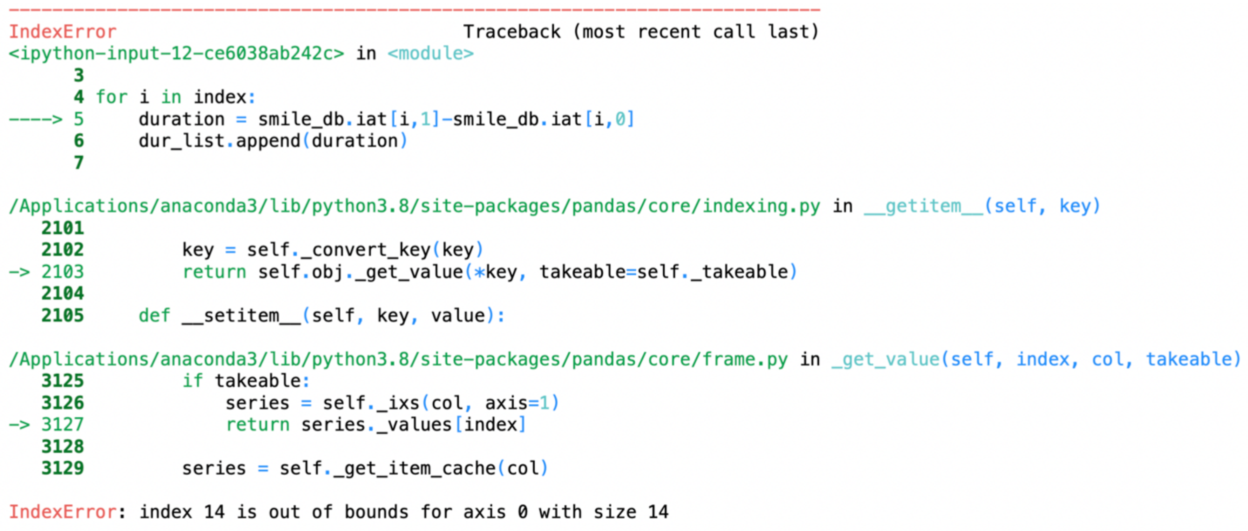
7) Provisional steps at the beginning of Session 2

1. Create csv file for one directory (smile/neutral)
2. Read txt file and extract timestamps, phonemes, labels
3. Populate database
4. Define functions that measures formants and populates database
5. Iterate through database
6. Test with next directory
7. Visualise the data (optional)

I researched libraries and ended up choosing *csv* for the initial data wrangling, *Pandas* for the final data output, and *Parselmouth* for the acoustic measurements. Parselmouth allows you call functions from *Praat* directly from Python without having to install Praat. Praat is a popular acoustic analysis programme. After reading documentation on both Parselmouth and Praat, I got a finished result that achieved my goal for the day, albeit not in a streamlined or efficient way. The resulting code can be seen in the "Session 2" notebook.

During this session, a common error I was coming across was the following:

Figure 2: *Common error during session 2*



```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-12-ce6038ab242c> in <module>
      3 
      4 for i in index:
----> 5     duration = smile_db.iat[i,1]-smile_db.iat[i,0]
      6     dur_list.append(duration)
      7 

/Applications/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in __getitem__(self, key)
   2101 
   2102         key = self._convert_key(key)
-> 2103         return self.obj._get_value(*key, takeable=self._takeable)
   2104 
   2105     def __setitem__(self, key, value):

/Applications/anaconda3/lib/python3.8/site-packages/pandas/core/frame.py in _get_value(self, index, col, takeable)
   3125         if takeable:
   3126             series = self._ixs(col, axis=1)
-> 3127             return series._values[index]
   3128 
   3129         series = self._get_item_cache(col)

IndexError: index 14 is out of bounds for axis 0 with size 14
```

This happened because I was constantly updating the variable smile_db. Sometimes, I forgot to do so and would be trying to run code on a previous version of the dataframe which had more rows than the current version. I would get this error when trying to iterate through indices that were no longer there. I corrected this by being more mindful of altering the original data frame.

Session 3

In session 3 I started a new notebook, where I used the code from the previous session to define functions that organised my programme in a better way. I decided to not turn my data into data frames until the end, as I found working with csv files easier.

I broke down my code into smaller steps and tried writing code that was more "readable" and better documented. This notebook ended up being the main notebook, where I would add the next steps. See "Final notebook" for the resulting code.

Session 4

In session 4, I moved on to the next step of my project (5), iterating through the directories and collecting all file names. This was the most straightforward part of this project, as I am familiar with working with dictionaries. The results of this session can be seen in the "Session 4" notebook.

Session 5

In this session, I moved the code from the previous session to my main notebook and defined the last few functions that would create the desired resulting output: a single data frame and csv file that contains all measurements for all audio files for the speaker. See "Final Notebook"

Session 6

I created a GitHub account and uploaded all notebooks, data folders and this written report for easy access.

Results and Bugs

In general, this project involved a lot of research as the data manipulations were more complex that I was used to. However, by end the end of the project I got familiar with the basic methods of the libraries I used.

In the end, the resulting programme only partially delivers on the desired goal. The following is a list of improvements needed to achieve this goal fully:

- Deleting consonants and silences
  Step 2.2 cleans the data extracted from the text files to only keep vowels. This is important because we are measuring Pitch and Formants, two features that can only be measured in vowels. Trying to measure these features during silences and consonant will result in invalid measurements. In the example included in the notebook, we can see several rows where the value for the column "Phoneme" is a consonant or a silence ("sil"). While step 2.2 should have taken care if this, it did not perform in the expected way.

- Concatenating all Data Frames
  The previous bug results in an error when trying to concatenate all data frames for the speaker. While I have tested get_neutral(), get_smile(), and speaker_data() independently, I have not been able to test that they all work together to create the final database.

- Running speed
  This programme is iterating multiple times through large datasets. While it seems to work generally, it takes a long time for the programme to run. I believe it could be done in a more efficient way with more advanced coding. This would be especially relevant if trying to use this programme to analyse data from several speakers. Improved efficiency would mean this programme can be used to analyse speaker data in bulk and obtain enough data to train AI models and more.