Project Proposal

Zoe Cummings

November 2024

1 Project Overview

In the SNaPP Lab at William and Mary I am a part of a team looking into the politicization of podcasts over time. After having solidified our research design and the scope of the project this semester, we have moved towards data collection. Our original data collection method involved recording the audio for the episodes we have picked from our sampling and then putting those through WhisperAI to get text transcriptions. Around half way through this semester we found that iHeartRadio had begun posting transcriptions for individual podcast episodes which opened the door for a new and more efficient data collection method. As I started working on creating the script to scrape the podcast information and transcriptions I realized I was often relying on the automation techniques and organization principles we had been learning this semester which is why I ultimately decided to continue my work on this data collection and storage for the final project. As the team stands right now I am the only Data Science major working on the creation of the code to scrape the iHeartRadio data, however, we are getting new data science team members next semester. As well as improving the efficiency of my data collection code I also wanted to have detailed instructions for future lab members on how this can be replicated moving forward. For the non data science members I decided to have a database with the information and the option to download the data as a CSV file would be beneficial for overall usage for a range of data science skills.

Before starting this work as a final project I had created a function that would take a link to an iHeartRadio page with the episode details and transcription, and scrape the text of the transcription. The work to automate the collection of the transcription links and and getting podcast information as well as the creation of the database are all new additions.

2 Data Description

The data used would be self collected using the scraping script I created. The script can only be used on podcasts from iHeartRadio that have transcripts available. The example podcast that currently in the scraping code is the Bobby Bones Podcast. The data will be collected and then saved as a JSON. The data

collected includes name of podcast, name of episode, data of episode, length of episode, and full transcription. This will be stored in the database as well. I will start by collecting data for a limited amount of podcasts so that the structure of the data collection can be well tested, as well as the process of adding data into the database, without making a significant amount of the work time be the data collection itself.

3 Methods

The **iHeartRadio** podcast web scraping tool ($\text{get}_t ranscriptions.py) consists of multiple Python functions. The$

3.1 get_links()

This function extracts all the links to podcast transcriptions by navigating through the podcast page and clicking the "See More" button to reveal additional episodes. This function takes the longest time to run as you need to wait for the pages to load while the computer presses the button and loads more data.

Parameters:

- `link`: The URL of the iHeartRadio podcast page (string).
- `pod_name`: The name of the podcast (string, used for file names).

This will scrape all episode links and save them to a file named `transcript_links_{pod_name}.txt` in the `json_files` folder.

3.2 get_transcription()

This function extracts transcriptions, episode titles, dates, and lengths (in minutes) from a specific podcast episode URL.

Parameters:

- `url`: The URL of the podcast episode (this must be a specific episode from iHeartRadio that has the transcript available).

This will return the episode's title, date, length, and the transcription text.

It can be called on its own with the URL being the link to an iHeartRadio page that has the transcription and information on a single episode of a podcast, or it can be called within the `process_links_from_file()` function.

3.3 process_links_from_file()

This function processes the links saved in a text file (`transcript_links_{pod_name}.txt`) saved in the `json_files` folder. This function takes the episode name, date, length, and transcription from the `get_transcriptions()` function and saves them in a nested dictionary structure through a JSON file.

Parameters:

- `filename`: The file containing the episode links (`transcript_links_{pod_name}.txt`).
- `pod_name`: The name of the podcast, which will be used to name the output JSON file.

Example of JSON structure:


```
{
  "Bobby Bones Show": [
    {
      "episode_title": "Tues Post Show: Lunchbox Makes An Empty Promise, DIY Dental Work And
      "episode_date": "December 17, 2024",
      "episode_length": "55",
      "transcription": "It's time for the Bobby Bones post show."
    }
  ]
}
```


3.4 complete_task()

This function automates the entire process: it extracts the episode links, scrapes the transcription data, and saves the results into a JSON file. It does this by calling the `get_links()` and `process_links_from_file()` functions.

This is the function I recommend using if you want to see the entire data collection process through. Note that you want to give yourself at least an hour to collect data as, by the nature of web scraping with a "See More" button, it is easiest to do it all at once, which is what this code supports.

Parameters:

- `start_url`: The URL of the overall podcast page.
- `pod_name`: The name of the podcast.

4 Results/Outputs

From this project I have developed a way to access transcription data for podcasts. Transcription data is not very accessible so further addition to this data will serve as a source for future analysis not only for the lab I am in but hopefully at large. From this project there are also two different formats to use the podcast data in, a JSON file and a database (.db) file. The snippet of the JSON file can be seen above for reference to the structure of the data. The database is available for download on the github repository to be used with sqlite. This is the table structure for the database: Podcast Database Structure

$\text{podcast}_{m}etadata - podcast_{n}ame - podcast_{i}d - pod_{d}escription - transcript_{e}xist - ideology - ideology_{i}episode_{m}etadata - podcast_{i}d - episode_{i}d - episode_{n}ame -$

$date - ep_description - lengthpopularity - podcast_id - rank - datetranscripts - podcast_id - episode_id - transcript$

A lot of time was spent on this project working on standardizing the web scraping so that it could work across different podcasts. Even with a significant amount of time put towards this I still run into issues with it where it will work for one podcast but stop working for another. After looking online this may be iHeartRadio recognizing a pattern of web scraping behavior. Either way I want to keep working on this to make sure it is a reliable data collection tool. Additionally I would like to flesh out the database by adding outside data from the Brooking's institute to show political ideology of podcasts, and scrape additional data from iHeartRadio on the popularity of podcasts over time. Achieving those goals would help with my overall motivation of the project to create a useful tool and data source for the SNaPP lab. Going further from that I want to continue collection of transcription data and additional outside data sources to create a more comprehensive look into podcast data. This could give way to interesting NLP analysis between podcasts as well as looking into individual changes in podcasts over time.

5 References

https://www.brookings.edu/articles/a-new-data-set-for-better-monitoring-of-the-political-podcast-ecosystem/