Allegheny College
Department of Computer Science

Senior Thesis

Music Recommendation by Mapping Music and Descriptive Paragraph

by

Xingbang Liu

ALLEGHENY COLLEGE

COMPUTER SCIENCE

Project Supervisor: **Janyl Jumadinova**
Co-Supervisor: **Gregory M. Kapfhammer**

April 6, 2020

Abstract

Provide a concise summary of your research project of approximately 250 words. Remember that the abstract is *not* an introduction, it is a *summary* of the entire document, including the results and future direction of the project.

Acknowledgment

Theses should acknowledge assistance received in any of the following areas:

- Designing the research
- Executing the research
- Analyzing the data
- Interpreting the data/research
- Writing, proofing, or copyediting the manuscript

Abbreviations

All abbreviations used in the thesis should be listed here, with their definitions, in alphabetical order. This includes trivial and commonly used abbreviations (at your own discretion), but not words that have entered into general English usage (such as laser or DNA). In particular, non-standard abbreviations should be presented here. This is an aid to the reader who may not read all sections of the thesis.

PPT	positive partial transpose
SRPT	Schrödinger-Robertson partial transpose

Glossary

Dipole Blockade	Phenomenon in which the simultaneous excitation of two atoms is inhibited by their dipolar interaction.
Cavity Induced Transparency	Phenomenon in which a cavity containing two atoms excited with light at a frequency halfway between the atomic frequencies contains the number of photons an empty cavity would contain.

If desired, an optional and short dedication may be included here.

Contents

List of Figures

List of Tables

Template Overview

You should first modify the documents in the preamble, things that appear before the main text as detailed below.

Front page: use the one provided in this template, after changing the values like names in the file `preamble/mydefinitions.tex`.

Abstract: There should be a single paragraph of about 250 words, which concisely summarizes the entire proposal, written in the file `preamble/abstract.tex`.

Acknowledgments, Abbreviations, Glossary, Dedication preamble pages are optional and can be used at the author's discretion.

The main text of the proposal should be stored in the "SeniorThesis.tex" document. The following descriptions are sections that must be included in the thesis document.

Bibliography: The bibliography should include all references cited in the text (as [**?**]) and it should not include references that have not been cited. ACM referencing style should be used when preparing the bibliography. We recommend using BibTeX or BibLaTeX and using the file `preamble/bibliography.bib`.

Chapter 1

Introduction

1.1 Motivation

It is normal to have the situation that, when people cannot find the right music track for certain emotional states.

1.1.1 Human Emotion and Music Emotion

Humans like music for its capability to express emotions. The study has shown that music can evoke human emotions and influence human moods [28]. In the experiments mentioned by Przybysz, some sad or happy music can cause human physical reactions such as muscle tension and hormone release. This is one of the reasons for the human to have certain emotions toward music. However, sometimes music emotion and human emotion are separate. Human and music may carry the same emotion, for example, sadness, but human sadness and music sadness may be different. The human may be seeking for temporary separation from the real world. Context outside of music could also arouse human emotions. For example, when people are at the funeral, they are meant to be sad. When the lyrics or the music title are sad, people would also feel sad.

As music is so important to humanity and the internet became the most important tool in society nowadays, music service companies started to provide people access to music content and convenient service bundles. Based on different user demands, different music services have different strategies. For example, some users want to access a large number of music tracks with a low cost, some users want to discover new music content in a smart way [27]. Therefore, Spotify provides music stream services, and users have the option to upgrade to Spotify premium. With Spotify premium, users can skip songs, make tracks offline with users options, and more. With a good music service, users use music playlist published by other users to play at coffee shops, users listen to music when exercising, and users play music at weddings.

1.1.2 Easy Music Discover

With a large amount of music content users can access, it would be hard for users to pick the right song at the right moment, expand their playlists, or listen back to the song they saved previously [27]. Therefore, some music service providers developed music

recommendation systems for users to find music content smartly. Spotify's Weekly Discovery is a good example for users to expand their playlist based on the music they saved previously. Generally speaking, the music recommendation system we can find on market works in two ways. The first way if to find new music content based on users' previous interested music. The second way is to find users who have similar actions and music preferences and cross recommend songs to each other. Since different users would not possibly have the same music playlist, it would be efficient to suggest new songs to different users.

These music recommendation services are accurate, users may enjoy it, but this system has a limitation. It only digs deeper for user interests, but it does not expand them. The reason is that the information for music recommender systems to use only contains user interaction toward music tracks. The system can classify and rank music tracks that people like or dislike, but they cannot get information about the user's emotional state.

1.1.3 User Current Emotion State

As mentioned before, the relationship between music emotions and human emotions are complicated. Not only some music can evoke human emotions, in some situations, but human emotions can also be evoked by the surroundings, lyrics, and humans may have their emotions and they are seeking similar emotions in music. In situations when people are sad, they want some corresponding sad music tracks to help them relieve mental painfulness. The recommendation system will recommend a list of songs based on user previously saved music. However, users would not only have sad music in their playlist, but they also have happy music. It would be hard for users to locate the songs they want. Although some music services have a sad music classification, users still have to explore more for the music that they want, because the scene defined by the lyrics may be different.

The important part here is to recognize the emotional state of the users. If music service recommends music content based on user current emotion state, the result would be more accurate in terms of user current demand. Therefore, it is better to let users express themselves. Let users define their emotions states themselves. This way, users can get music recommendations more accurate because users know their requirements the best. This solution can also provide a wider range of recommendations because it does not depend on user interaction history. Every music content suggested by this theory would be potentially new from users' past preferences.

1.2 Current State of the Art

There are various music services, such as Spotify, Amazon Prime Music, Pandora, and YouTube Music, that people like to use. The most popular music service on the market would be Spotify. According to MIDiA [24], a company analyzes media and technology, Spotify owns 83 million subscribers and 36% of the market share [26] at H1 of 2018.

1.2.1 Spotify

Spotify has different clients on different platforms. For example, there are mobile clients, desktop clients, TV clients, Web player, and more. Users can choose freely between different platforms. In each Spotify platform, the user can choose their preferred music content to play [37]. With smart home technologies got more and more popular, Spotify added functions that can control playback devices on different platforms. For example, users can control Google Home to play music from Spotify by smartphones. Besides the basic music player function, users can also add other users as friends to get other people's music content.

Spotify also has a strong music recommendation system to help people get more new music that they like. Users can get recommendations from Discover Weekly, Daily Remix, and song radios. There are three main techniques Spotify used to get recommendations for the users, which are Collaborative Filtering, Natural Language Processing, and Audio Metadata Modeling. In other words, Spotify utilized information from user interactive history and compares across different users, lyrics semantics, music patterns, and audio metadata, to get similar music contents.

1.2.2 YouTube Music

YouTube Music is another popular music service on the internet nowadays. YouTube was famous for its video services. YouTube users can upload videos they produced. Users who want to enjoy personalized services have to register user accounts. Based on user interests, YouTube would recommend videos on their homepage, and under each video that been played by the users, several related videos would be shown for users to pick. Under each video, users can write comments to share their thoughts. Video providers can also interact with other users. Video producer not only can gain popularity on YouTube but also money from YouTube because YouTube let companies put advertisements in the video contents. This is one of the reasons people are willing to upload videos to YouTube.

YouTube got more and more popular in recent years, they have announced that they have 100 hours of videos uploaded to them every hour in the past [23]. According to a user insight report in 2018, about 47% of users listen to music on-demand at Youtube [9]. With such a good user foundation, music producers and companies started to release music videos on YouTube. Based on a large number of licensed music videos, YouTube started music streaming services. With a good video recommendation system, YouTube combined content recommendation and social media functions. Similar users are easier to get each other's suggestions.

1.2.3 Netease Cloud Music

Netease Cloud Music is a Chinese music service provider. Most of the music license is held by Tencent Music at the time. As a new music service provider, to compete with Tencent Music, Netease Cloud Music developed stratify to develop strong music recommendation systems and social media platforms.

They think the essence of music service is to help users find the music they like

efficiently. When users find the music they like, they would usually want to express themselves. Therefore, Netease Cloud Music provides comment sections for users to write their feelings and stories. Users can also share music between friends and other social media platforms. Because of the lack of music license, Netease Cloud Music encourages small and new producers to upload music to their platform. They also depend more on music recommendation between similar users. Due to Netease Cloud Music's good reputation among users, they got a great number of fundings recently [33].

1.3 Goals of the Project

A new system was proposed to match descriptive paragraphs with music tracks. This way, users can express their emotions whenever it is necessary, and a complex mechanism to detect the user's current emotion state can be avoided. Besides simple emotion detects, they can also expand their music interest by exploring more random but relevant music contents.

1.3.1 Scenes

Multiple scenes would be fit for the project. For example, a person breaks up with his or her girlfriend or boyfriend would be very sad at the moment. People would listen to sad songs when that happens. To find the right music tracks to identify their exact feelings, the person who is experiencing negative emotions can write literature paragraphs. These paragraphs would contain the sentiment of that person, and the story behind the negative emotions. It would be the same for people experiencing positive emotions, such as fall in love with someone.

The scene does not limit to self-generated emotions. For example, a person who watches a film or television could find himself or herself related to the work. The person would listen to related music tracks to express appreciation. Or the filmmakers want to find relevant music that matches the scene of the film, even contains matching metaphor. They can use a description of the film or the film scripts as the descriptive paragraph. By mapping the key factors of the paragraph with the music key factors, the relevant music tracks would be identified.

1.3.2 General Workflow

Before the user uses the service, pre-processing is required to get extra music contexts. The lyrics of the music track would be analyzed, and they would be summarized into keywords. And the result would be stored at the server database.

After the pre-processing is done, this system would firstly take user inputs. However, users are lazy, it always requires more to motivate people to generate sufficient inputs. Therefore, the format of the input has to be closely related to potential emotion breakouts. These inputs can be a diary, personal blog, video script, or description of the video content.

After getting a sufficient amount of user inputs, the system would analyze the inputs, breaks them down into pieces and summarize them into keywords. The sentiment of the text would also be analyzed. This extracted information would be used to compare with the language model in the music dataset. Specifically, the similarity would be checked between user description summarization and music fingerprints.

Finally, the system will return a list of relevant songs to users. The general workflow of the system shown as figure 1.1.

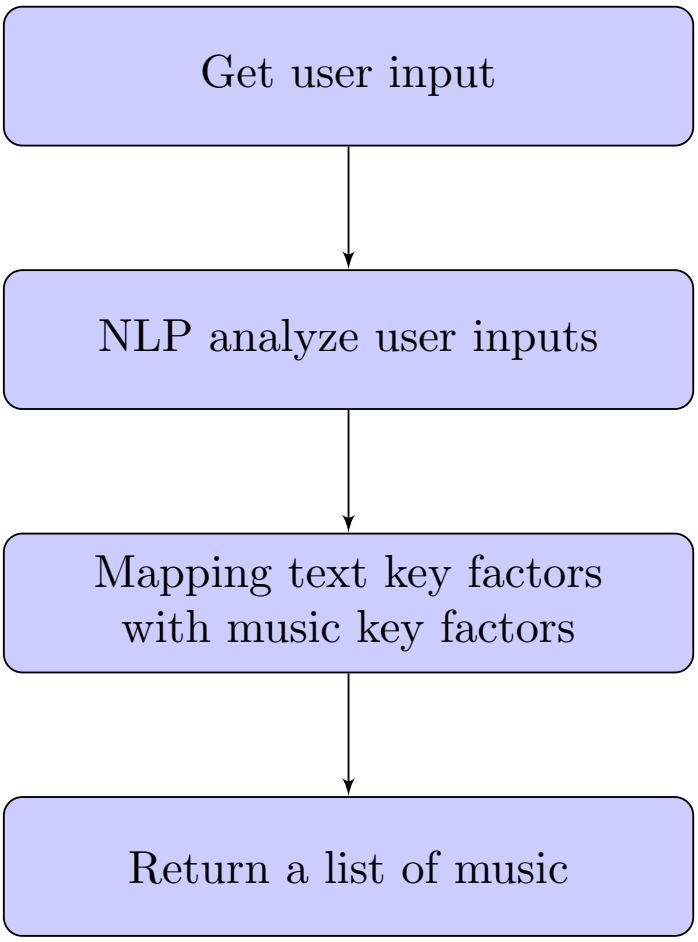


Figure 1.1: General Workflow

1.4 Thesis Outline

In this project paper, five sections will be used to introduce and analyze the proposed music recommendation system.

- The introduction will be mainly talking about the motivation of the project. Popular music services will also be introduced and analyzed. Besides motivation and popular music services, the expected goal of the music recommendation system and the general process steps will also be explained.

- The related work section will be mainly introducing similar studies. Each study will be introduced and analyzed toward their progress and limitations.
- The method section will be mainly about how is the system being build. Packages and tools used to build the system will also be introduced.
- The experiment section will be mainly talking about the evaluation of the system. The evaluation process and final results will be introduced and analyzed.
- Finally, the key findings and general ideas will be wrapped up in the conclusion section.

Chapter 2

Related Work

As a popular entertainment business, music service was always been a popular topic in the industry. Both the recommendation system and music emotions have many existing research projects.

2.1 Spotify Recommendation System

A recommendation system or recommender system is an algorithm that can suggest user-preferred items to a user by giving score or probability to items [32]. In the music recommendation system, the item to suggest would be music.

Take Spotify as an example, it has three parts in their recommendation system, which are Collaborative Filtering, Natural Language Processing, and Audio Metadata Modeling.

2.1.1 Collaborative Filtering

Collaborative Filtering [35] is an algorithm that recommends new content between similar users or items. The algorithm predicts user preferences by learning from users' past ratings for items [15].

Item-based

The first way to do collaborative filtering is to predict the relationship between items. Therefore, calculate item similarity is the first step. The common ways of calculating similarity for item i and j, sim(i,j), are cosine similarity, Pearson correlation, adjusted cosine, and conditional probability.

Let the targeted users be set U, $r_{u,i}$ and $r_{u,j}$ be user's rating over item i and item j. The cosine similarity can be calculated by equation (2.1).

$$\text{sim}(i,j) = \cos(\mathbf{i},\mathbf{j}) = \frac{\mathbf{i}\cdot\mathbf{j}}{\|i\| * \|j\|} = \frac{\sum_{u\in U} r_{u,i} r_{u,j}}{\sqrt{\sum_{u\in U} r_{u,i}^2}\sqrt{\sum_{u\in U} r_{u,j}^2}} \tag{2.1}$$

Let $\bar{r}_u$ be the average rating for user u. The adjusted cosine can be calculated by equation (2.2) [2].

$$\text{sim}(i,j) = \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{u \in U} (r_{u,j} - \bar{r}_u)^2}} \tag{2.2}$$

Let $\bar{r}_i$ be the average rating for item i. The Pearson correlation can be calculated by equation (2.3). One good thing about correlation similarity is that it can calculate how close are the items related [15].

$$\text{sim}(i,j) = \frac{\text{Cov}(i,j)}{\sigma_i \sigma_j} = \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_i)^2} \sqrt{\sum_{u \in U} (r_{u,j} - \bar{r}_j)^2}} \tag{2.3}$$

Finally, the conditional probability can be calculated by equation (2.4).

$$\text{sim}(i,j) = P(j|i) = \frac{f(i \cap j)}{f(i)} \tag{2.4}$$

After obtained the similarity between items, a value of the item for the targeted user can be predicted. It means to obtain how the user rates similar items for specific item i, the record we obtained could is the history records.

User-Based

Another way to do collaborative filtering is to find the users that are similar. After knowing the similar users, and the item similarity scare are obtained, the recommendations can be predicted.

Take Spotify as an example, after collecting the user interaction, for instance, the playlist, whether liked the song, a unique coordinate will be generated for each user. The collected fields are the dimensions. The item-based suggestion will be coming from analyzing the user playlists, and the user-based suggestion will be coming from comparing different users. Once the similar users are identified, the algorithm will simply recommend contents that one user has and others do not.

2.1.2 Natural Language Processing

Natural language processing is the study of the interaction between computers and human languages. Human language can be speeches or writings. This study has many sub-fields, for example, speech recognition, text summarization, and text generation. By using natural language processing techniques, the meaning of lyrics can be classified, which can be further used to identify the music genre. The sentiment of the lyrics can also be analyzed, which can be used to identify the mood of the music tracks.

Text Sentiment Classification

Text classification is a very important sub-field of natural language processing, many real-life applications are depending on text classification. For example, web search, document classification, and information ranking [20]. The classification can be based

on different aspects, in music recommendation examples, sentiment can be a feature. Sentiment analysis is the contextual mining of text emotions, for instance, positive or negative. Happiness, joyful, and compliment emotions can be categorized as positive emotions. Sad, angry, and guilt can be categorized as negative emotions. The general text classification workflow is shown in figure 2.1.

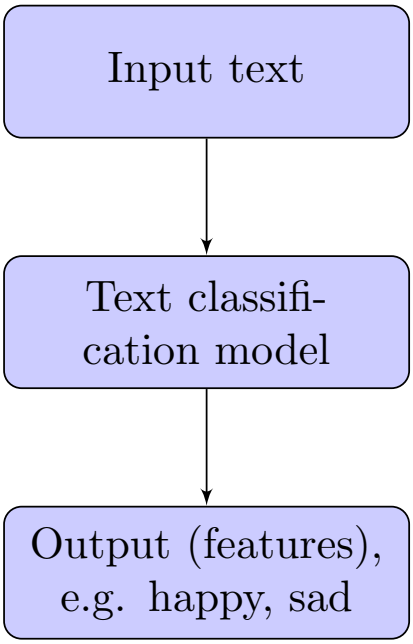


Figure 2.1: Text Classification Workflow

The first step to classify text is to treat text as a bag of words [13]. Bag of words is a model in natural language processing that treats texts as unordered words, and the frequency of each word would be recorded. This process is usually conducted by tokenization, and followed by stemming and lemmatization. Stemming and lemmatization can delete the additional part of the word, remain only the stem part. For example, after stemming and lemmatization, the word "interesting" would become "interest". After stemming and lemmatization, stop words, which are words that have no actual meaning in a sentence, will be removed from the bag of words. An example of a stop word can be "and".

One way to classify texts is a Naive Bayes algorithm. Naive Bayes is based on Bayes' probability theorem [31]. In Bayes' probability theorem, for event A and B, the conditional probability of A given B is equation (2.5).

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.5}$$

Multinomial naive Bayes is one of the ways to use the Naive Bayes algorithm to classify texts. It uses the term frequency, $tf(t, d)$, where t is term, and d is the active document. Let $tf(t, d)$ be the frequency of term t in document d, and n_d be the total number of terms in document. As the equation (2.6) shows.

$$\text{term frequency } = \frac{tf(t, d)}{n_d} \tag{2.6}$$

Spotify uses textual information from the internet to find key terms for music content. The terms will be used to build vectors for each song. A similar technique used in collaborative filtering will be applied to find similar content. This technique has a flaw that the textual information gets from the internet is too broad. It will be better if the information is from the comments under each song directly. That way, each song will have a direct correlation to the key terms. Except for comments, user search history could also be used as part of the text [14].

2.1.3 Audio Metadata Modeling

Audio metadata is the information for the audio files, it usually contains names for artist, album, title, genre, track number, and more [10]. After sampling the audio, more information can be included in audio metadata, for example, loudness, tempo, and more. This information can be used to predict potential music or human emotions. Tempo represents the speed of the audio, it is related to human emotions according to studies. Fast tempo and emotions like joy and fear have correlations, slow tempo and emotions like sadness and tenderness have correlations [21].

Spotify has a database that contains the audio metadata. These aspects can be used to build vector representations. By applying a similar technique in collaborative filtering, similar content will be identified. Bogdanov et al. proposed a way of classifying music by using audio metadata [12]. They asked participants to choose their preferred music and categorized them based on their preferences. They then modeled the audio files and classified them. After that, they correlated the classified features with predefined categories. By using this model, they can suggest songs based on audio metadata.

2.2 Content-Based Music Information Retrieval

Besides collaborative filtering, content-based music information retrieval is another effective way of recommending. This method collects information that describes the music, and use the information to suggest new content. Audio metadata can be one of the sources for the information.

One application is to collect and classify audio signal features, then apply them to match based on different situations. In general, there are two ways of matching, which are query by example, and query by humming [22]. Query by example means getting the audio signal as input and return music metadata as output. This query method works on the specific music track, but not the variation tracks, for example, a cover version. Query by humming makes it up by getting melody as input and returns similar tracks. This application is more of a searching or matching algorithm. The limitation of this recommendation method is as I mentioned previously, lack of novelty [22]. It only expands on user saved songs, and it cannot predict user situation.

The other application is to combine the information which describes the music with user preferences. The user preference does not mean the user ratings, content-based music recommendation does not rely on user ratings [15]. However, categorizing music content based on descriptive information requires experts of the fields to set

rules. A similarity check is also required for the content-based suggestion. The way of calculating the similarity between two items is through calculating their distance. The common ways of calculating such distance are Euclidean distance (2.7), Manhattan distance (2.8), vector cosine distance(2.1), Mahalanobis distance (2.9), and Chebyshev distance (2.10).

$$d = |x - y| = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2} \tag{2.7}$$

Euclidean distance measures the straight line between two points in N-dimensional space [6].

$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i| \tag{2.8}$$

Manhattan distance is also a measurement of the straight line between two points in N-dimensional space. Just imagine the street blocks in Manhattan, there two paths from one intersection to the other one. However, the true distance between them is a straight line cut through the buildings [17].

$$d(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)} \tag{2.9}$$

Mahalanobis distance is a measurement of the straight line distance between two points in N-dimensional space. Mahalanobis distance handles better with correlated points(≥ 2). In situations that there are more than 2 points that are correlated, the axes of the points are no longer orthogonal, and they cannot be plotted in 3-dimensional space [34].

$$d(x, y) = \max_{i=1...n} |x_i - y_i| \tag{2.10}$$

Chebyshev distance, also called maximum value distance, is similar to Euclidean distance and Manhattan distance. It calculates the straight line distance based on point coordinates.

There are limitations for this application when new users entered the system, they can't get the right suggestion because the system needs time to adjust to the user preferences. This situation is also called the cold-start problem [15]. Another issue is feature extraction. It is hard to extract high-level descriptions such as mood. These high-level descriptions are very important in accurate and personalized recommendations.

2.3 Contextual Music Retrieval

Contextual music information is any information, other than music content information, that describes the music content itself. Contextual music suggestion means to recommend music based on the user's actual situation, for example, emotional state. This method takes three sources of information, environment-related context, user-related context, and multimedia context [22]. Environment-related information can be

the season, temperature, time, and weather because all these contexts can influence human emotions. User-related information can be user activities, such as exercising and driving, user profiles, such as social network statements, and emotional state. Multimedia can be text and image relevant to the music. This information can be used to profile and predict the user's actual situation. After the necessary information is gathered, similarity predicts, classification, and other techniques can be used to predict user preferences.

Another way of achieving contextual music retrieval is proposed by B. Han et al [18]. They proposed an emotion state transition model which models music-evoked human emotions, and content-based music recommendation ontology to profile user preference and context. The system provides music by mapping high-dimensional music features with the emotion state transition model. According to the paper, this method achieved 67.54% overall accuracy. The limitation of this method is that it needs a large amount of data, and the human emotional state can be changed based on the social environment. Once the social context is changed, more data would be required to get higher accuracy.

Recommendation based on contextual music retrieval is still new, and it has great potential to contribute to the music recommendation system.

Chapter 3

Method of Approach

This project is about providing a new concept of recommend user potentially interested music. Although it is only a munimal viable propotype, it would be good to make it a web application hosted on a website. This chapter contains the tools utilized for this application and the workflow for how do these tools work with each other.

3.1 Tools Utilized

The tools utilized for this application come from multiple areas. The general areas can be catgorized with front-end area and back-end area. The following sub-sections are the tools utilized.

3.1.1 Flask

Flask served as both front-end and back-end tool, it is a micro web application framework, which means it does not require other third party libraries and services. It is a stand along framework that you and add other parts onto it [8]. Flask was implemented by Python, which is a great fit for this application because this project processes a big amount of textual data. Python is famous for its simple syntax and abundant of libraries, which means it is a good fit for big data processing. Simple syntax means the code would be easy to read, and the processing flow would be easily translated into actual code.

Flask Front-end

Flask uses HTML templates to render the front-end pages. Any values wants to be changed dynamically when running the website can be passed into HTML files by using variables. For example, in route functions, add the variable in return statement as 3.1 shows. Then go to the redered HTML file, add 3.2 in the corresponding place.

Listing 3.1: Flask Route Template Variables

```
return url_for(
  "html_file_name",
  outgoing_variable_name = incoming_variable_name,
```

```
)
```

Listing 3.2: HTML Python Variables

```
{{outgoing_variable_name}}
```

In general, HTML templates are the frame of the pages, Flask controls the dynamic variables, and redering stratigies. In HTML files, *form* were used to collect user answers. As the code 3.3 shows.

Listing 3.3: HTML Forms

```
<form method="POST">
  Title of text: <input type="text" name="title">
  <br>
  <textarea name = "text" rows="25" cols="80"></textarea>
  <br>
  <input type="submit" value="Submit">
</form>
```

Flask Back-end

In flask, modules can be imported by using Python importing statements. All the setups can be put inside *__init__.py* file for initialization. For example, in 3.4.

Listing 3.4: Python Project Initialization

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
from flask_login import LoginManager

# App Configurations
app = Flask(__name__)
# secret key to protect request and cookie
# use
# import secrets
# secrets.token_hex(16)
# to generate token.
app.config["SECRET_KEY"] = "__keys__"
# utilize db
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///site.db"
db = SQLAlchemy(app)
bcrypt = Bcrypt(app)
login_manager = LoginManager(app)

from music_main import routes
```

After imorted and seted up all the necessary libraries, they can be used as normal Python programs. Other tools used in back-end will be explained in other sections of this chapter.

3.1.2 SQLAlchomy

With such big amount of data, the best way of saving and retrieving data is by using databases. Flask SQLAlchemy is a Flask extension that simplifies the implementation of Structured Query Language (SQL) [7]. To create database by using Flask SQLAlchemy, first initialize the library 3.5, then define the database as models 3.6. Use 3.7 to create database.

Listing 3.5: Flask SQLAlchemy Utilization

```
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///site.db"
db = SQLAlchemy(app)
```

Listing 3.6: Flask SQLAlchemy Model

```
class User(db.Model, UserMixin):
    id = db.Column(
      db.Integer,
      primary_key=True,
    )
    username = db.Column(
      db.String(20),
      unique=True,
      nullable=False,
    )
    password = db.Column(
      db.String(60),
      nullable=False,
    )
    input = db.relationship("Input",
    backref="author",
    lazy=True,
    )

    def __repr__(self):
        return f"User('{self.username}')"
```

Listing 3.7: Flask SQLAlchemy Create Database

```
db.create_all()
```

Relational Database

This project used relational database to manage user data, lyrics data, and processed data. The reason to choose relational database over non-relational database is that the data used in this project is strictly structured. When comes to trtriving resyults, it would be easier to use strictly defined tables. The data used in this project can be devided into three tables. The user table stores usernames, passwords, and its relation to other user data. The lyrics table stores the title of the song, the singer of the song,

the lyrics of the song, and the Summarization of the lyrics. The input table stores the title of the user text, the user inputed paragraphs, the Summarization of user input texts, and the top five results of each text matching processing. The table design and their relations is as the graph 3.1 shows.

Figure 3.1: Database Design and Relation

3.1.3 Lyrics API

All the lyrics this project used were coming from genius.com. Genius is a website provides music music knowledge such as lyrics and story behind music [1]. The developer website provides API to access their music knowledge.

LyricsGenius

LyricsGenius is a tool to retrive lyrics from genius.com on GitHub that was written in Pyhton [19]. This tool utilized Genius API to retrive song lyrics, but user have to register Genius developer accout and then register for application to get client access token. To get the lyrics, use *search_song* function to get a Python object, then access the lyrics attribution. As the code 3.8 shows.

Listing 3.8: LyricsGenius Sample Code

```
import lyricsgenius

genius = lyricsgenius.Genius("my_client_access_token_here")
song = genius.search_song(song_title, song_artist)
song_lyrics = song.lyrics
```

Although this tool will return None when there is no song found accrding to the search term, but the server would return vague results. Therefore, regular expression was used to filter out these vague results.

Playlists

To get a list of song titles and song authors, a website called playlist-converter.net [3] was used to convert saved Spotify playlist into csv format files 3.9. Then the file will be imported as Python list objects and will be further fed into LyricsGenius tool. In this project, a total of 4 playlists with 593 pop songs was added to the database. The name of the Spotify playlists are: *Happy Songs (80s, 90s, 2000s, 2010s & 2020s _ Best Happy Hits & Oldies*, *POP Music Playlist - Best POP Hits of All Time (Updated in 2020)*, *Sad Songs*, and *Top Pop*.

Listing 3.9: Spotify Pliaylist

```
"track","artist"
"Twist And Shout - Remastered","The Beatles"
"Uptown Funk (feat. Bruno Mars)","Mark Ronson"
```

```
"A–Punk","Vampire Weekend"
"Hooked on a Feeling","Blue Swede"
"Come","Jain"

...
```

Language Detection

For now, the only language model used was English, so, song language other than English will also be filtered. Language detection tool, spacy-langdetect, based on Spacy was used to detect whether the song lyrics is English [11]. To use the tool, Spacy and Spacy English language model should be installed. Then, use the Spacy and spacy-langdetect pipeline to create the object. As the code sample 3.10 shows.

Listing 3.10: Spacy Language Detection

```
python –m spacy download en

import spacy
from spacy_langdetect import LanguageDetector

nlp = spacy.load('en')
nlp.add_pipe(
  LanguageDetector(),
  name='language_detector',
  last=True,
)
doc_check = nlp(text)
  if doc_check._.language["language"] == "en":
      return True
```

3.1.4 Text Summarization

To filter out the redundant texts and retain only the useful texts, a Python libray called pytextrank was used to summarize each text entries. Pytextrank was implemented based on Spacy pipline, and Mihalcea 2004 paper [25], it can rank the phrases and sentences [16]. Based on the ranking, the top phrases will be used to formulate the summarized sentences. To use pytextrank, first install the dependencies, which are Spacy, NetworkX, and GraphViz. Then download the Spacy English model. After add textrank into the Spacy pipeline, import the text into the pipeline object. Finally, use *summary* function to get the Summarization of the text. As the sample code 3.11 shows.

Listing 3.11: Pytextrank Sample Code

```
python –m spacy download en_core_web_sm

import spacy
```

```
import pytextrank

nlp = spacy.load("en_core_web_sm")
tr = pytextrank.TextRank(logger=None)
nlp.add_pipe(tr.PipelineComponent, name="textrank", last=True)
doc = nlp(text)
whole_sent = ""
sum = doc._.textrank.summary(limit_phrases=15, limit_sentences=5)
for sent in sum:
  whole_sent = whole_sent + repr(sent).rstrip() + "␣"
```

3.1.5 Text Matching

There are many text maching algorithms, for example, Levenshtein, Jaccard index, and cosine similarity. Although they all compare text entries, each algorithm only has good proformance in certain areas. In situations like spelling check or keyword search, Levenshtein would be a good fit. Levenshtein algorithm used Levenshtein distance, it measures the minimum character edit steps from one string to another. Similar algorithms are Hamming and Jaro-Winkler. They are called edit based similarities. Edit based is simple and they are very effecient in short strings. However, they do not consider the meaning of the strings. Algorithms such as Jaccard index and cosine similarity are called token based similarities. The token here is a unit of the string, for example, n-gram. They are effecient in long strings, and they take the meaning of the text into account. However, they are not very effecient in short strings. These algorithms are good fit for areas such as text mining and bioinfomatics.

This project used a hybrid algorithm called Monge-Elkan. It combines both edit based similarities and token based similarities. It combined the advantages of edit based similarities and token based similarities. However, they can be slower than other similarity check algorithms. The tool this project used is called py_stringmatching. It has a module to compute the Monge-Elkan similarity of two strings [36].

To use py_stringmatching, first import the library. Then initialize the *MongeElkan* function. Finally, use the *get_raw_score* function to compare two string entries. As the sample code 3.12 shows.

Listing 3.12: Monge-Elkan Similarity

```
import py_stringmatching as sm

me = sm.MongeElkan()
return me.get_raw_score(user_sum, lyrics_sum)
```

3.2 Implementation and Workflow

This section explained how does the project organized and how were the tools used to construct the application. The workflow of the project is also explained in this section.

3.2.1 Project Implementation

This subsection explained the prerequisites, library management, and file structure of the project.

System and Python Versions

This project was implemented and deployed on Linux system. Specifically Ubuntu 18.04.4 LTS 3.13.

Listing 3.13: System Version

```
cat /proc/version
Linux version 4.15.0-76-generic (buildd@lcy01-amd64-029)
(gcc version 7.4.0 (Ubuntu 7.4.0-1ubuntu1~18.04.1))
#86-Ubuntu SMP Fri Jan 17 17:24:28 UTC 2020
```

The Python version used was Python 3.7.6 3.14. To implement the project with the desired Python version, Pyenv was used to avoid version conflict between installed version of Python with system versions of Python. Pyenv is a simple but powerful Python version management tool [29]. This tool can be used to download desired versions of Python easily. The scope of the Python can be set to global, which means the desired version of Python can be used system-wise, or local, which means the desired version of Python can only be used in a specific project.

Listing 3.14: Python Version

```
pyenv versions
  system
* 3.7.6 (set by /home/sheldon/.pyenv/version)
```

Library Management

Many tools and plugins was used in this project. To avoid the dependency confliction, Pipenv was used to manage the packaging. Pipenv is a dependency management tool [30]. With pipenv, all the specification can be recorded in *Pipfile* 3.15. For example, Python version, dependencies, customized scripts.

Listing 3.15: Pipenv Pipfile

```
[[source]]
name = "pypi"
url = "https://pypi.org/simple"
verify_ssl = true

[dev-packages]

[packages]
lyricsgenius = "==1.8.2"
spacy = "==2.2.4"
networkx = "==2.4"
```

```
graphviz = "==0.13.2"
pytextrank = "==2.0.1"
numpy = "==1.18.2"
pip = "*"
six = "==1.14.0"
py-stringmatching = "==0.4.1"
spacy-langdetect = "==0.1.2"
attrs = "==19.3.0"
bcrypt = "==3.1.7"
beautifulsoup4 = "==4.6.0"
blis = "==0.4.1"
catalogue = "==1.0.0"
certifi = "==2019.11.28"
cffi = "==1.14.0"
chardet = "==3.0.4"
click = "==7.1.1"
coverage = "==5.0.4"
cymem = "==2.0.3"
decorator = "==4.4.2"
idna = "==2.9"
importlib-metadata = "==1.5.0"
itsdangerous = "==1.1.0"
langdetect = "==1.0.7"
more-itertools = "==8.2.0"
murmurhash = "==1.0.2"
plac = "==1.1.3"
pluggy = "==0.13.1"
preshed = "==3.0.2"
py = "==1.8.1"
pycparser = "==2.20"
pyparsing = "==2.4.6"
pytest = "==5.4.1"
requests = "==2.23.0"
srsly = "==1.0.2"
thinc = "==7.4.0"
tqdm = "==4.43.0"
urllib3 = "==1.25.8"
wasabi = "==0.6.0"
wcwidth = "==0.1.8"
zipp = "==3.1.0"
Flask = "==1.1.1"
Flask-WTF = "==0.14.3"
Flask-SQLAlchemy = "==2.4.1"
Flask-Bcrypt = "==0.7.1"
Flask-Login = "==0.5.0"
Jinja2 = "==3.0.0a1"
```

```
MarkupSafe = "==1.1.1"
SQLAlchemy = "==1.3.15"
Werkzeug = "==1.0.0"
WTForms = "==2.2.1"
gunicorn = "*"

[requires]
python_version = "3.7"

[scripts]
music = "./scripts/music.sh"
setup = "./scripts/setup.sh"

[pipenv]
allow_prereleases = true
```

With Pipenv, Python vertual environment can be set up and denpencies with users' required version can be install easily and recorded into Pipfile by using one line of bash command. If there are any confliction between denpendencies, Pipenv can also help to diagnose the confliction and provides resolving advises. A complete list of requirements in the requirement file is as follows 3.16:

Listing 3.16: requirements.txt

```
-i https://pypi.org/simple
attrs==19.3.0
bcrypt==3.1.7
beautifulsoup4==4.6.0
blis==0.4.1
catalogue==1.0.0
certifi==2019.11.28
cffi==1.14.0
chardet==3.0.4
click==7.1.1
coverage==5.0.4
cymem==2.0.3
decorator==4.4.2
flask-bcrypt==0.7.1
flask-login==0.5.0
flask-sqlalchemy==2.4.1
flask-wtf==0.14.3
flask==1.1.1
graphviz==0.13.2
idna==2.9
importlib-metadata==1.5.0 ; python_version < '3.8'
itsdangerous==1.1.0
jinja2==3.0.0a1
langdetect==1.0.7
```

```
lyricsgenius==1.8.2
markupsafe==1.1.1
more-itertools==8.2.0
murmurhash==1.0.2
networkx==2.4
numpy==1.18.2
packaging==20.3
plac==1.1.3
pluggy==0.13.1
preshed==3.0.2
py-stringmatching==0.4.1
py==1.8.1
pycparser==2.20
pyparsing==2.4.6
pytest==5.4.1
pytextrank==2.0.1
requests==2.23.0
six==1.14.0
spacy-langdetect==0.1.2
spacy==2.2.4
sqlalchemy==1.3.15
srsly==1.0.2
thinc==7.4.0
tqdm==4.43.0
urllib3==1.25.8
wasabi==0.6.0
wcwidth==0.1.8
werkzeug==1.0.0
wtforms==2.2.1
zipp==3.1.0
```

File Structure

The project repository contains the application directory. This directory has the HTML templates, application Python files, and the database file. The playlist directory contains all the Spotify playlists. The scripts directory contains the script to set up the environment and download the dependencies automaticlly then start the service. Test directory contains all the testing program for the application. Texts directory contains all the input files wrote by the users. In the project root directory, there are app main function file, Pipfile, Piplock file, and requirements record text file.

The file structure of this project is as follows 3.2:

3.2.2 Workflow

This section explained the deployment steps of this project, workflow of the process, and the user access flow.

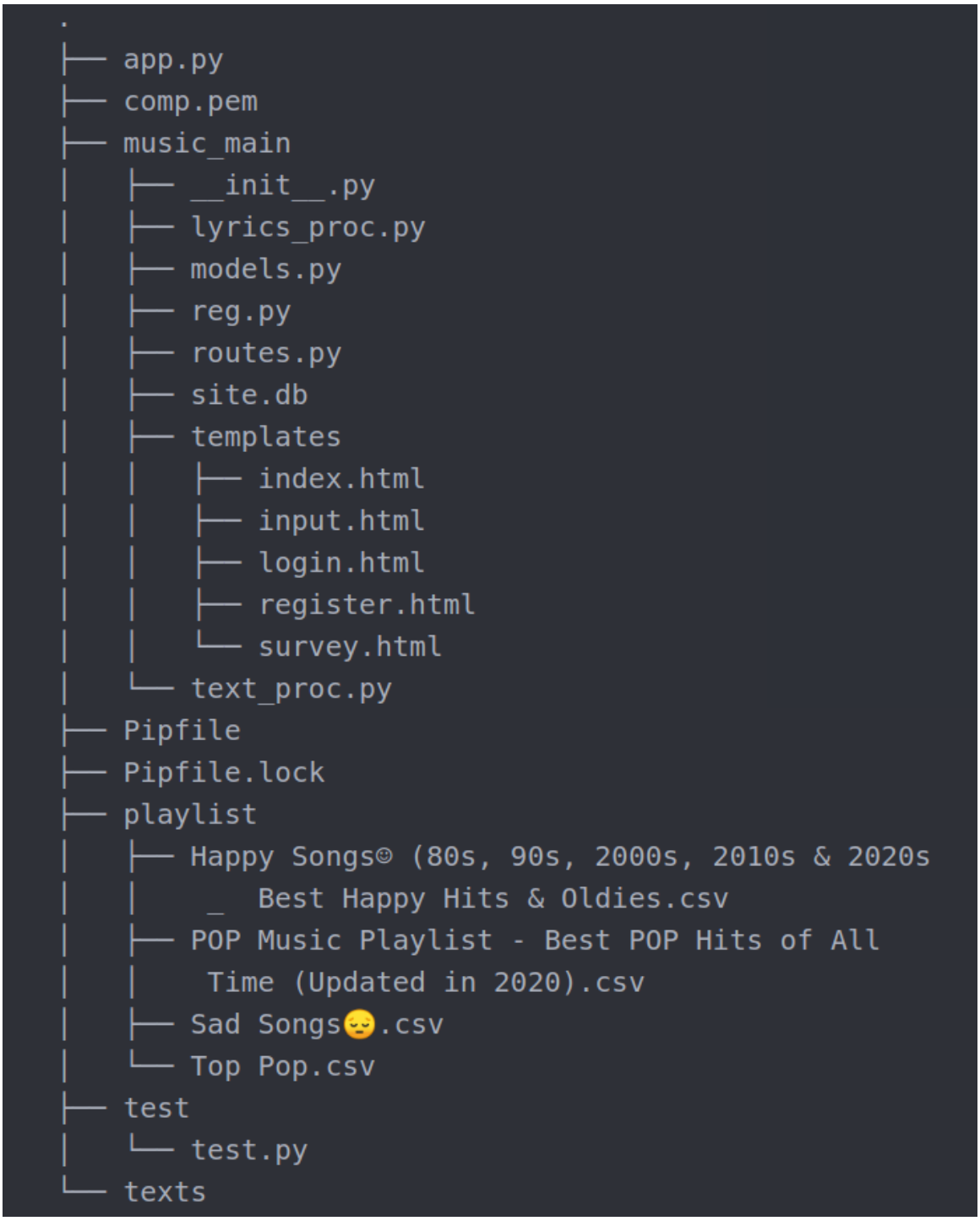


Figure 3.2: File Structure Tree

Deployment

After finised the implementation, the project Pipfile was locked, which means all the dependency versions was locked. To automate the setup process on new environment, a script was added to the project. Under script section of Pipfile, add the command keyword and the path of the script, for example, *setup = "./scripts/setup.sh"*. The command to setup a new environment would be *pipenv run setup*. In the setup script, there were command to download the two English language models, from Spacy, the command to create the databases, and the command to run the Lyrics model to update the Lyrics table with lyrics from Genius, and summarization of each lyrics. The Bash script is as follows 3.17:

Listing 3.17: setup.sh

```
#!/bin/bash

# Download language model
pipenv run python -m spacy download en_core_web_sm
pipenv run python -m spacy download en
# create database
pipenv run python -c "from music_main import db; \
from music_main.models import User, Input, Lyrics; \
db.create_all()"
# Create data for Lyrics table
pipenv run python -c \
"from music_main.lyrics_proc import *; lyrics_main()"
```

After the above preperations, ssh into the server and clone the repository from GitHub. Install the Pyenv and then the version of Python used to implement the Project. Install the Pipenv, and use *pipenv install* to create Python vertual environment and install all the denpendencies according to the Pipfile. Nginx and Gunicorn was used as web servers during deplyment on Amazon AWS server. Supervisor was used to moniter the application process on the Amazon AWS Linux server.

Workflow

When everything is ready and the application is up and running on the server, the URLof the website can be shared to people who are interested in participating the experiment.

When lyrics model was ran during the deployment process, the playlist was first combined into one list. Then the list was loop through and was used to get the lyrics from Genius by using lyricsgenius library. After making sure the result was real lyrics and the language was English by using regular expression and spacy-langdetect, the lyrics was summarised by calling summarization function powered by Pytextrank from text model. Finally, the corresponding information was added and committed to the database table.

After user filled in their response, the text was saved as text files, and the summarization function was called from the text model. Then the text matching function

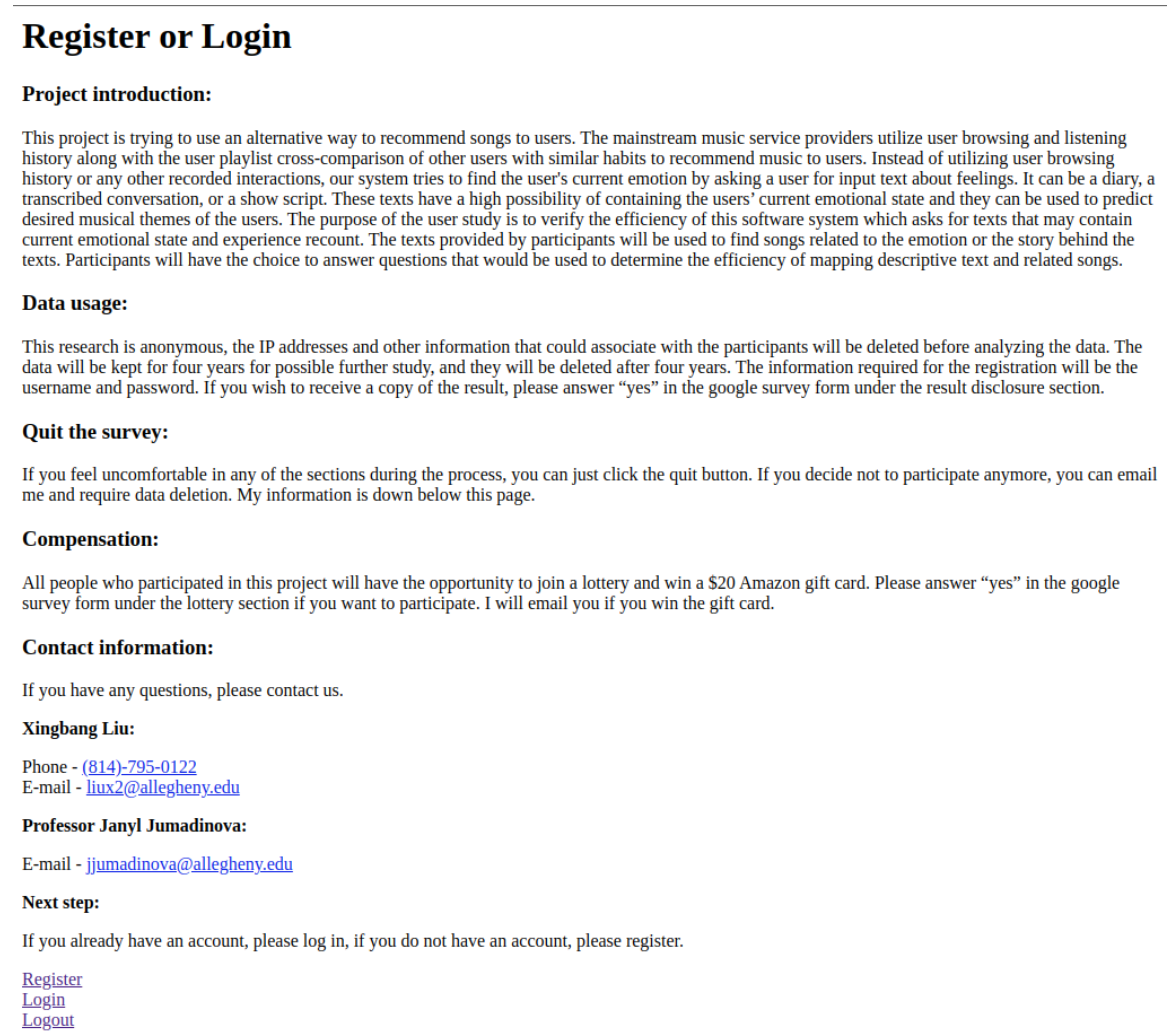
powered by py-stringmatching was also called from the text model. The top 5 results was kept, and all the corresponding information was saved into the database table.

The complete system design is as graph 3.3 shows.

Figure 3.3: System Design

Access Flow

When users access the website to use the application and to participate the experiment, the landing page will contain the brief project Introduction, data usage, how to quit the experiment anytime, compensation lottary participation, experimentor contact information, log in link, log out link, and register link. As the graph 3.4 shows. After user registered and logged in, user will be landed in the user input page 3.5. Here user will read the instruction and fill out the blanks. In the result page, a list of no more than 5 songs will be shown to the user, followed by the link to the survey and compensation participation sign up link 3.6. The sitemap of this application is as graph 3.7 shows. The Walk-Through section under Experiments also talked about the access flows.



Register or Login

Project introduction:

This project is trying to use an alternative way to recommend songs to users. The mainstream music service providers utilize user browsing and listening history along with the user playlist cross-comparison of other users with similar habits to recommend music to users. Instead of utilizing user browsing history or any other recorded interactions, our system tries to find the user's current emotion by asking a user for input text about feelings. It can be a diary, a transcribed conversation, or a show script. These texts have a high possibility of containing the users' current emotional state and they can be used to predict desired musical themes of the users. The purpose of the user study is to verify the efficiency of this software system which asks for texts that may contain current emotional state and experience recount. The texts provided by participants will be used to find songs related to the emotion or the story behind the texts. Participants will have the choice to answer questions that would be used to determine the efficiency of mapping descriptive text and related songs.

Data usage:

This research is anonymous, the IP addresses and other information that could associate with the participants will be deleted before analyzing the data. The data will be kept for four years for possible further study, and they will be deleted after four years. The information required for the registration will be the username and password. If you wish to receive a copy of the result, please answer "yes" in the google survey form under the result disclosure section.

Quit the survey:

If you feel uncomfortable in any of the sections during the process, you can just click the quit button. If you decide not to participate anymore, you can email me and require data deletion. My information is down below this page.

Compensation:

All people who participated in this project will have the opportunity to join a lottery and win a $20 Amazon gift card. Please answer "yes" in the google survey form under the lottery section if you want to participate. I will email you if you win the gift card.

Contact information:

If you have any questions, please contact us.

Xingbang Liu:

Phone - (814)-795-0122
E-mail - liux2@allegheny.edu

Professor Janyl Jumadinova:

E-mail - jjumadinova@allegheny.edu

Next step:

If you already have an account, please log in, if you do not have an account, please register.

Register
Login
Logout

Figure 3.4: Landing Page

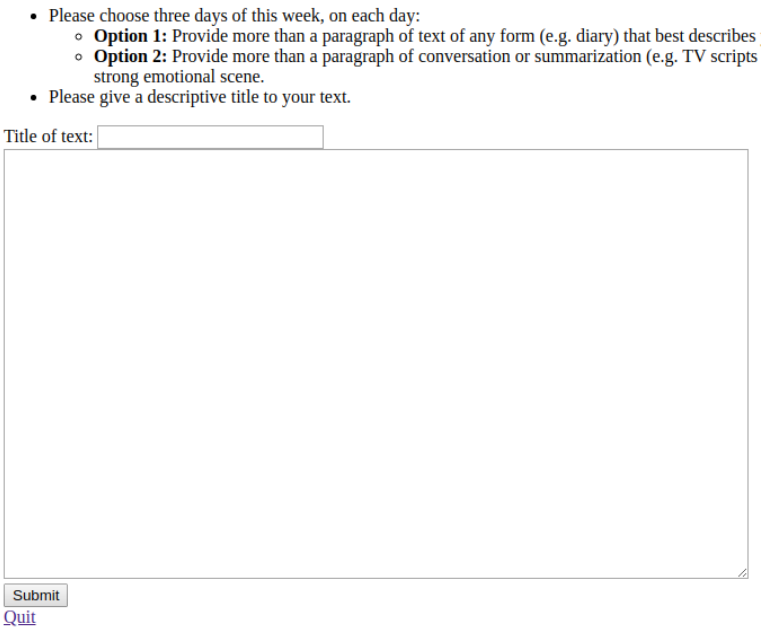


Figure 3.5: Input Page

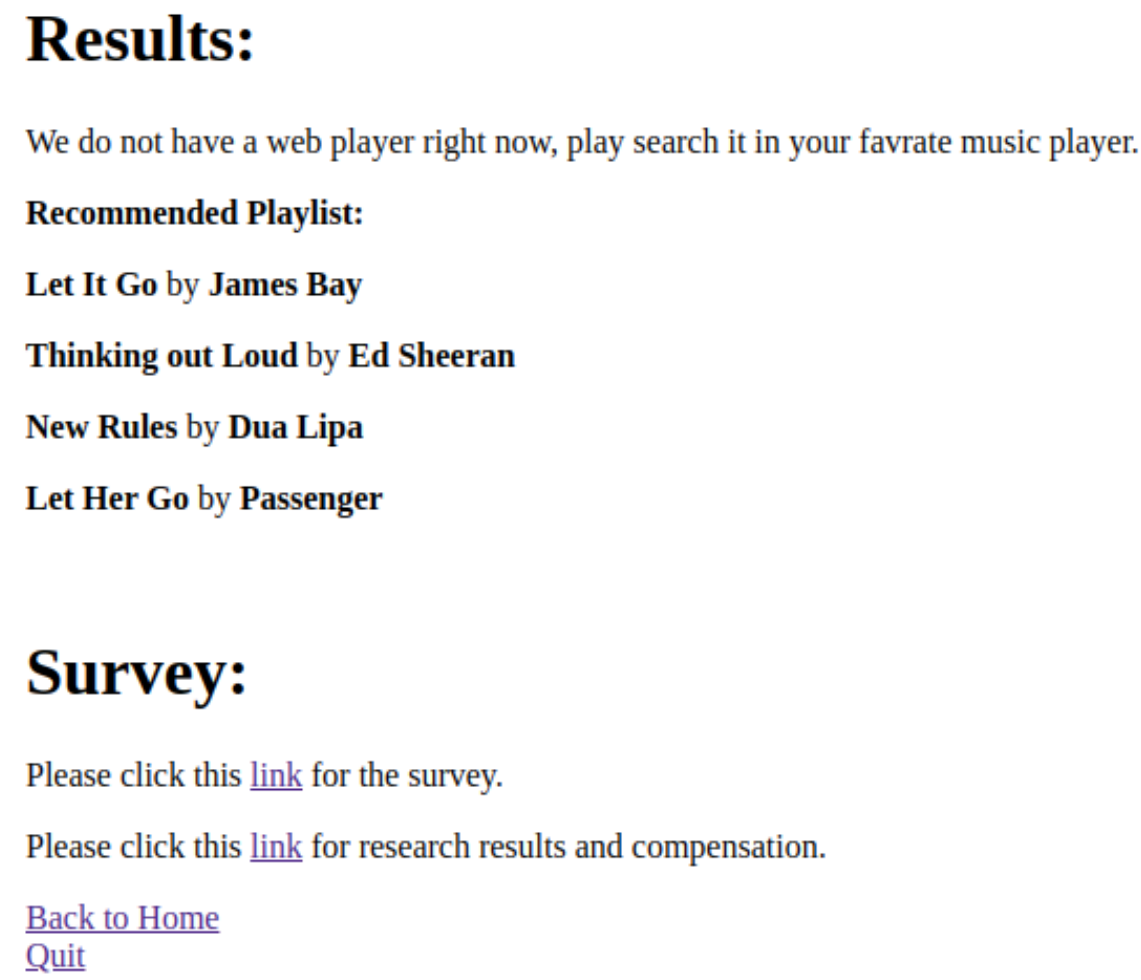


Figure 3.6: Result Page

Figure 3.7: Sitemap

Chapter 4

Experimental Results

4.1 Experimental Design

4.1.1 Program Tests

4.1.2 User Tests

Institutional Review Board

The form of user tests are surveys. To conduct the user servey, the designed survey have to be approved by Allegheny College Institutional Review Board (IRB). Institutional Review Board is an organization that reviews and supervises experiments to make sure the human rights of participants do not get hurt [4]. Before the application starts, experimentors have to finish a Collaborative Institutional Training Initiative (CITI Program) [5]. The program required by the Allegheny College Institutional Review Board is Social Behavioral Research. To get approved by the IRB, a proposal first have to be submitted. The proposal contains review questions to help experiment designer and organizers to identify the type of experiment because certain types of experiment can get exemption or expedition. The questions are, for example, does the experiment involves human, does the experiment involves individuals under the age of 18. This experiment required exempted review. Other than the review type, a brief introduction, procedure, and walk-through should be included.

Description and Rationale of Project

The description and rationale of project included in the Institutional Review Board proposal are as follows:

This develops a novel software system to recommend songs to users. The mainstream music service providers utilize user browsing and listening history along with comparison of the user's playlist to that of other users with similar habits to recommend music to users. In this project, instead of standard data, our system uses machine learning to identify the user's current emotion by asking a user to enter a text that contains some indication of their current emotions and feelings. The text entered by the user can be their diary, a transcribed conversation, or a script from a show that their current emotional state relates to. Using this text, our system then learns about

the users' current emotional state and uses that information to predict desired musical themes and songs to the user. The purpose of the user study is to verify the efficiency of this software system, where the participants are asked to provide a short body of text, and following the review of the recommended song list they are invited to provide feedback on the outcome they received.

Methods and Procedures

The methods and procedures of the experiment included in the Institutional Review Board proposal are as follows:

The participation for this study will be advertised on My Allegheny, and via departmental announcements (posters, club emails, Slack channel). The individuals who agree to participate in the study will be emailed a link to the website on which our system operates. Once on the landing page of the website, the participants will be provided information about the experiment and they will be informed that their consent is received if they proceed with the next steps of the study. In the next steps, the participants will create an account in our system and then log in to their unique account. The walk-through section contains detailed sequence of steps that the participants will be involved in and the pages on our website that they will be directed to.

The website will be hosted on a private cloud-based server (Amazon Cloud Server, AWS). The IP addresses and other information that could associate with the participants will be deleted before analyzing the data. The data will be kept for four years for possible further study, and it will be deleted after four years. AWS's data centers and network architecture is designed to protect information, identities and applications stored in their cloud by following core security and compliance requirements. The information required for the registration will be the username and password. After logging in, the participants will be asked to follow the instructions on the website and provide a body of text that reflects their emotions on three different days. The following instructions will be provided:

> Please choose three days of this week, and for each day:
>
> **Option 1:** Provide more than a paragraph of text in any form (e.g. diary) that best describes your emotions or experiences on that day.
>
> **Option 2:** Provide more than a paragraph of a conversational or summarization text (e.g. TV scripts or TV scene summarization) that describes an emotional scene.
>
> Please provide a descriptive title to each body of text that you enter.

After receiving the bodies of text from the user, our system will automatically provide a list of recommended songs. After getting the recommended music playlist, participants will be invited to participate in the survey by answering the following three questions:

1. Describe the body of text that you provided. What emotions did you want to express in this text?

2. Please choose the best description for the recommended playlist.
   (a) This playlist has nothing to do with my text
   (b) I can see a rare relation to my text
   (c) This playlist is acceptable, but I do not have a strong emotional connection to it
   (d) I have a fair amount of an emotional connection to this playlist
   (e) I have a strong emotional connection to this playlist
3. If you feel an emotional connection to this playlist, why do you think the music is related to your provided text? If you do not feel a connection to this playlist, what do you think the playlist is missing?

The goal of the questions in the survey above is to assess the efficiency of the algorithm and the participants' interpretation of their feelings. The answers will be collected on the same website of our system and the answers will be connected to specific user accounts. The user account is the only connection between the answers and the provided bodies of text by the user. Again, all the user information, including IP address, browser information, usernames, and passwords will be deleted before the analysis of the collected data.

After the survey, the data will be analyzed using the responses in the user satisfaction from question 2. Responses in the range from 1 to 3 will be marked as "need for further analysis". Question 1 and question 3 will be used to interpret the unsatisfied playlist. The summary of the analyzed results will be reported in a senior thesis document that will be stored on the cloud-based version-control system, called GitHub. All participants who indicate an interest in receiving the results of the study will be emailed a link to the GitHub page containing the senior thesis document.

The potential risks of this study is that the participants may spend a long time writing texts. They may also have to think about sad memories. If the participants do not wish to proceed, they may quit any time during the process by pushing the *quit* button on the page.

Walk-Through

Participants will receive a link to begin their participation in this user study. When participants click on the link, they will land on the "Information" page.

Information Page After entering the website, users would see an information page that contains the following message:

> **Project introduction:**
>
> This project develops a novel software system to recommend songs to users. The mainstream music service providers utilize user browsing and listening history along with comparison of the user's playlist to that of other users with similar habits to recommend music to users. In this project, instead of standard data, our system uses machine learning to identify the user's

current emotion by asking a user to enter a text that contains some indication of their current emotions and feelings. The text entered by the user can be their diary, a transcribed conversation, or a script from a show that their current emotional state relates to. Using this text, our system then learns about the users' current emotional state and uses that information to predict desired musical themes and songs to the user. The purpose of the user study is to verify the efficiency of this software system, where the participants are asked to provide a short body of text, and following the review of the recommended song list, participants are invited to provide feedback on the outcome they received.

Data usage:

This research is anonymous, the IP addresses and other personal information that could be associated with the participants will be deleted before analyzing the data. The data will be kept securely on Amazon Web Services for four years for possible further study, and they will be deleted after four years. The information required for the registration is the username and password. If you wish to receive a copy of the results of the study, please answer "yes" in the google survey form under the result disclosure section. A copy of the report of the study in the form of the senior thesis project document will be sent to you upon the completion of the analysis.

Exiting the study:

The potential risks of this study is that the participants may spend a long time writing texts. They may also have to think about sad memories. If you feel uncomfortable in any of the sections during the process, you can click the quit button. If you decide not to stop the participation, you can email the principal investigator of the project at *liux2@allegheny.edu* to request immediate deletion of any data you have created.

Compensation:

All participants will have the opportunity to join a drawing for a chance to win a $20 Amazon gift card. Please answer "yes" in the google survey form under the lottery section if you want your name to be entered into the drawing. All participants will receive a notification of the results of the drawing.

Contact information:

If you have any questions, please contact us.

Xingbang Liu:

Phone: (814)-795-0122

E-mail: liux2@allegheny.edu

Professor Janyl Jumadinova:

E-mail: jjumadinova@allegheny.edu

If you consent to the participation in this study, please continue with the next step.

After participants log in, they will be taken to the "Text Entering" page.

Text Entering Page After participants enter their text according to the instruction, they will be directed to the "Result" page containing their recommended playlist, below which information about the survey will be included.

Result Page After listening to the songs recommended, users can proceed to the servey.

> **Survey 1:** Please click on this link for the survey to provide feedback on your recommended playlist.
>
> **Survey 2:** Please click on this link to indicate your interest in receiving the results of this study and your interest in entering the drawing.

4.2 Evaluation

4.3 Threats to Validity

Chapter 5

Discussion and Future Work

This is the conclusion. You might want to leave it unnumbered, as it is now. If you want to number it, treat it like any other chapter.

This chapter usually contains the following items, although not necessarily in this order or sectioned this way in particular.

5.1 Summary of Results

A discussion of the significance of the results and a review of claims and contributions.

5.2 Future Work

5.3 Conclusion

Bibliography

[1] About genius.

[2] Adjusted Cosine Similarity.

[3] Convert your music playlists.

[4] Institutional review board.

[5] Mission and history.

[6] 'n'-dimensional euclidean distance.

[7] Sqlalchemy¶.

[8] Welcome to flask¶.

[9] IFPI releases 2018 Music Consumer Insight Report, Oct 2018.

[10] Metadata in digital audio files – what it is, where it is and how to tidy it up., Jan 2018.

[11] abhijit2592. A fully customisable language detection pipeline for spaCy, May 2019.

[12] Bogdanov, D., Haro, M., Fuhrmann, F., Xambó, A., Gómez, E., and Herrera, P. Semantic audio content-based music recommendation and visualization based on user preference examples. *Information Processing & Management 49*, 1 (2013), 13–33.

[13] Brilis, S., Gkatzou, E., Koursoumis, A., Talvis, K., Kermanidis, K. L., and Karydis, I. Mood classification using lyrics and audio: A case-study in Greek music. In *IFIP Advances in Information and Communication Technology* (2012).

[14] Bu, J., Tan, S., Chen, C., Wang, C., Wu, H., Zhang, L., and He, X. Music Recommendation by Unified Hypergraph: Combining Social Media Information and Music Content. In *Proceedings of the 18th ACM international conference on Multimedia* (2010).

[15] Celma, Ò. *Music Recommendation and Discovery*. 2010.

[16] ceteri. Python impl for TextRank, June 2017.

[17] Craw, S. Manhattan distance, Jan 1970.

[18] Han, B. J., Rho, S., Jun, S., and Hwang, E. Music emotion classification and context-based music recommendation. *Multimedia Tools and Applications* (2010).

[19] johnwmillr. LyricsGenius: a Python client for the Genius.com API, Dec. 2019.

[20] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).

[21] Kamenetsky, S. B., Hill, D. S., and Trehub, S. E. Effect of tempo and dynamics on the perception of emotion in music. *Psychology of Music 25*, 2 (1997), 149–160.

[22] Kaminskas, M., and Ricci, F. Contextual music information retrieval and recommendation: State of the art and challenges, 2012.

[23] Liikkanen, L. A., and Salovaara, A. Music on YouTube: User engagement with traditional, user-appropriated and derivative videos. *Computers in Human Behavior* (2015).

[24] MIDiA-Research. About Us, 2019.

[25] Mihalcea, R., and Tarau, P. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (2004).

[26] Mulligan, M. Mid-Year 2018 Streaming Market Shares, Sep 2018.

[27] Palaniappan, V. Spotify Product Analysis, Jan 2018.

[28] Przybysz, P. Music and emotions. *Avant* (2013).

[29] Pyenv. Simple Python Version Management: pyenv, Jan. 2019.

[30] Pypa. Pipenv: Python Development Workflow for Humans, Nov. 2018.

[31] Raschka, S. Naive bayes and text classification I - introduction and theory. *CoRR abs/1410.5329* (2014).

[32] Ricci, F., Rokach, L., and Shapira, B. Introduction to recommender systems handbook. In *Recommender systems handbook*. Springer, 2011, pp. 1–35.

[33] Russell, J. China's NetEase raises $600M for its music streaming business, Nov 2018.

[34] Stephanie. Mahalanobis distance: Simple definition, examples, Aug 2018.

[35] Su, X., and Khoshgoftaar, T. M. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence* (2009).

[36] Team, W. M. A comprehensive and scalable set of string tokenizers and similarity measures in Python, Feb. 2019.

[37] Zhang, B., Kreitz, G., Isaksson, M., Ubillos, J., Urdaneta, G., Pouwelse, J. A., and Epema, D. Understanding user behavior in Spotify. In *Proceedings - IEEE INFOCOM* (2013).