Software Development Proposal

Music recommendation system

----“OffScore”

Group Member:

Wenhe Xu 1098619

Zuoyang Fu 1098502

Yihang Liu 1098329

Professor: Dr. Pinata Winoto

Date: 2022/2/19

1. Introduction

Music recommendation systems can be treated as an advanced tool to provide a powerful recommendation function for users. Common music recommendation systems usually read users' log streams and usage data to implement the music recommendation function, which can not combine the environment to give the song that users want to hear. Therefore, to improve the current music recommendation systems, our group hopes to apply the AI algorithms based on the detection of Beat Per Minute to develop the precise recommendation services and provide users with songs that fit their environment and mood.

2. Project Description

The core functionality of this system is to implement the precise music recommendation by interacting with real-time data from users. Firstly, users of the system will wear the T-Wristband smart bracelet, which measures the user's heart rate, location and other data in real time through an embedded program. At the same time, the client on the cell phone forms an information transmission chain with the bracelet via Bluetooth. This software can generate a music song list for the client in real time that matches the current environment through a sophisticated recommendation algorithm, and transmit it back to the client's bracelet to provide reference information. The formation of efficient and real-time information communication between the user's bracelet and the software is an important technical difficulty of this software.

3. Requirement

3.1. Hardware requirement

This hardware is a wearable device, which can measure the user's heart rate, location and other data in real time through an embedded program.

3.1.1 LILYGOTTGO T-Wristband & DIY ESP32 smart bracelet.

3.1.2 LILYGOTTGO Pulse Detection Module. As an attachment to T-Wristband, the pulse detection module allows the band to receive instant information of the user's pulse and display on the screen. It also supports further extensive functionality like bluetooth transfer.

3.2. Software requirement

3.2.1 Music Player.

A general open source player is sufficient for application, and it needs to have certain decoding capabilities of MP3, WMV, FLAV and other streaming formats. Allows external calls as well as running in the background. Allow song switching, volume control, and creating playlist. The project can accept two player development schemes:

- Localization: Use the existing GitHub open source player.
- Invocation: Use Netease Cloud API: https://github.com/Binaryify/NeteaseCloudMusicApi

3.2.2 Bluetooth Transmission.

Contains two operation patterns, from bracelet to the terminal and vice versa.

- Bracelet to the terminal: manually switch the song signal, manually pause the signal, heart rate.
- Terminal to bracelet: music information.

3.2.3 Music Tag System.

Assign additional tags to the user-specified range of music (consider upgrading to the online version), the tag content includes:

- BPM (Use MixMeister BPM Analyzer to batch)
- Scene
- Style

These tags are independent of attributes common to music formats, such as creator, artist, date modified, duration, encoding format, etc.

3.2.4 Embedded Database.

The database is built on a cloud server and can use any relational or non-relational database. See the "**Database**" entry below for details.

3.2.5 Algorithm execution module.

A scripting class written in a programming language to perform music recommendations based on a database and received signals. Its working principle is to generate a dynamic playlist, that is, the next song to be played. When the user's data tends to a stable range, the songs played should not change. For example, if the heart rate is maintained between 85-90 and the change is not large, the playlist will not be refreshed until the next song is played.
In addition, the algorithm execution module also needs to call the music player. After receiving the manual song cutting instruction, or after the current song is played, it should immediately execute the operation of playing the next song. The history of playlist playback can be saved by generating a log stream, and if necessary, it can be developed as an advanced recommendation algorithm, or support the operation of switching songs to the previous song.

3.2.6 App Permissions:

1. Positioning
2. Turn Bluetooth On & Off
3. Read and write mobile phone storage
4. Media volume control
5. Get body movement information

3.3 AI Algorithm - Recommendation Algorithm

The basic recommendation algorithm is based on heart rate and location (based on satellite positioning) to find music corresponding to the tag.
The advanced recommendation algorithm can assign different influence factors according to the user preference vector (latent factor) and the result vector of the basic algorithm to jointly decide the next song selection. The user preferences and user portraits here can refer to the following:

1. Labeled recommendation: Recommendation based on labels requires products to have label concepts at the beginning. According to the

previous manual label delineation, user labels can be obtained by processing UGC content and data analysis of user behavior.

2. Mixed Weighted recommendation, the weighted summation of different recommendation algorithms and influencing factors can be used to mine the relationship between users and their favorite music in more detail, thereby improving the recommendation accuracy.

$$f(u,i) = \lambda_1 s_1(u,i) + \lambda_2 s_2(u,i) + \ldots + \lambda_n s_n(u,i) + \sum_{j=1}^{n} \lambda_j s_j(u,i,e_1,e_2,\ldots e_k)$$

λ refers to different weight coefficients, s2(u,i) represents different recommendation algorithms, and ek represents other factors that affect the user's preference for listening to songs (such as the user's weather, geographic location, time).

If the development cycle is limited, the task is to complete the basic recommendation algorithm.

4. Databases

4.1. Classficaion

A. Beat Per Minute（BPM）:

- 50~ 85 BPM（Depressio/Sadness）
- 90~105BPM (Introverted)
- 110~125BPM(joyful)
- 130~150BPM (excited)
- >150BPM mania

B. Suitable scene: city, campus, home, countryside, park

C. Music Style: classical, pop, rick, blues, hop-hop, jazz, country music

4.2 Data Collection and Storage

Collect the relevant data from the Million Song Dataset. Each song corresponds to a file, and the fields include information about the songs, such as BPM, Artist_name, Suitable_scene, Music_style, etc. All the data will be stored in the MySQL database for retrieval and querying.

5. Budget

Devices	Price /¥	Link
T-Wristband H303	201	LILYGOTTGO T-Wristband
T-Wristband Heart Rate Module	28	LILYGOTTGO T-Wristband

6. Process

a) Engineers should complete the task of the software requirement analysis. We need to write the outline analysis and detailed design specifications of the software, as well as recommendations for database and algorithm selection.
b) Considering the specific requirements of this project, the developer needs to select a suitable bracelet as the hardware to support the data stream during the design process.
c) Software development can be done through agile development by breaking down the entire software into multiple modules and then designing the modules. The primary task in this is to determine the program flow, algorithms and data structure, and the secondary task is to design the database, and the common method is the structured programming approach.
d) Last but not least is the software testing phase, where developers can find as many bugs as possible with minimal cost before the product goes live. The common testing methods used in general are the white-box and black-box testing methods of testing source programs.

7. Reference

[1] Analysis of NetEase Cloud Music Recommendation Algorithm.
https://zhuanlan.zhihu.com/p/63908049
[2] Work Principle of NetEase Cloud Music Playlist Recommendation Algorithm.
https://www.jianshu.com/p/d40ce25bec21