A MUSIC PLAYER WITH SONG SELECTION FUNCTION FOR A GROUP OF PEOPLE

Jun'ichi Suzuki and Tetsuro Kitahara
College of Humanities and Sciences, Nihon University, Tokyo 156-8550, Japan
{junichi,kitahara}@kthrlab.jp

ABSTRACT

There are often situations in which a group of people gather and listen to the same songs. However, majority of existing studies related to music information retrieval (MIR) have focused on personalization for individual users, and there have been only a few studies related to MIR intended for a group of people. Here, we present an Android music player with a music selection function for people who are listening to the same songs in the same place. We assume that each user owns his/her favorite songs on his/her Android device. Once a group of users gathers each user can launch this player on his/her smartphone. Then, the player running on each device starts to communicate with other devices via Bluetooth. Information about songs stored in every device, along with the playback history, is collected to a device referred to as the master device. Then, the master device estimates each user's preference for every song based on playback history and music similarity. The master device then extracts songs that are highly preferred and sends a command to start playback to the devices storing these songs. Our experimental results demonstrate the successful estimation of music preferences based on music similarity.

1. INTRODUCTION

In situations such as parties and carpooling, it is common that a group of people gather and listen to the same background music. However, it is not easy to select songs in such situations: if a particular member selects songs based on his/her musical preferences, other members with different musical preferences may be unsatisfied. To resolve this problem, we need a mechanism to extract each member's musical preferences and select songs taking into account those preferences.

Although music information retrieval (MIR) has a long history of technology development [3], relatively few attempts have been made to develop MIR techniques for a group of people. MusicFX, developed by Jseph et al. [6], selects a music broadcasting station from 91 stations specializing in different music genres, and the selection process is indirectly influenced by the musical preference of each member present at that place. This system uses a preference database consisting of every member's preference for 91 genres on a scale of -2 to 2. After detecting who is present based on each member's electronic badge, the system determines a broadcasting station using this database with a group preference arbitration algorithm. This algorithm basically computes a group preferences as a squared summation of individual preferences. Crossen et al. [4] developed a similar system called Flytrap. This system also attempts to play songs that are pleasing to every person present. The system detects the people who are present and sends information about each member's previous music choices to a server. Based on a voting mechanism in which high votes are given to those that have been listened to previously, songs to be played back are determined. A web application developed by Popescu et al. [8], called GroupFun, helps a group of friends agree on a common music playlist. With GroupFun, users can listen to and rate their own songs as well as their friends' songs. The system then arbitrates between the users' preferences using four different algorithms to determine which songs to play. BlueMusic, developed by Mahato et al. [1], uses Bluetooth to send an indivdual's musical preference data to a public music playback system. Individual users enter their musical preferences into a web form in advance and obtain strings, such as "Bm+A1R3EST3," that encodes their musical preferences. Their mobile devices then broadcast these data as Bluetooth device names. The public music playback system collects such strings to determine which songs to play back. In addition, some researchers developed music recommendation systems for a group of people [5,7]. These systems typically assume that (1) information on individual users' musical preferences is collected (or estimated from playback histories) in advance and (2) songs to be played back are stored in a server or public playback system.

Here, we develop a music selection and playback application under the following assumptions:

- Individual users own songs inside their own smartphones (or mobile devices).

  This means that songs to be played back are stored separately in multiple devices. Appropriate songs should be selected from a music collection dispers-

© Jun'ichi Suzuki and Tetsuro Kitahara. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jun'ichi Suzuki and Tetsuro Kitahara. "A Music Player with Song Selection Function for a Group of People", 18th International Society for Music Information Retrieval Conference, Suzhou, China, 2017.

ing at multiple devices and should be played back without manually switching any settings.

- No server or special equipment is necessary.

  The application should run on individual users' devices and songs are never copied to a server to avoid copyright issues.

- An individual device possesses information about only its playback history.

  The application running on each device has no prior knowledge about how much the device owner favors each song stored in other devices. The appliaction has to estimate this information from data that the device has.

To meet these assumptions, we design the application based on the following policy:

- One of the devices is regarded as the master devices, and every device communicates the list of songs stored in it and its playback history (in particular, how many times the device owner has played back each song) to that device.

- The master device does not collect the waveforms of songs stored in other devices but rather commands the device storing a song to be played back to connect itself directly to the Bluetooth speaker. If a song stored in a different device is selected next, this device will be automatically connected to the speaker after the current device is disconnected from the speaker.

- Each user's preference for songs stored in others' devices is estimated based on the similarity to songs stored in his/her own device.

The rest of the paper is organized as follows: In Section 2, we describe the overview of our application and present a method for estimating the degree of preference noted above. In Section 3, we report the system implementation and experiments. Finally, we conclude the paper in Section 4.

2. SYSTEM OVERVIEW

This application aim to select songs from a collection separately stored in different devices and play the songs back seamlessly. As discussed in the Introduction section, we designed the application based on the following policies:

- Every device communicates information about songs (not the waveforms themselves) to the master device.

- After the song selection process occurs, the master device commands the device storing the selected song to connect itself directly to the Bluetooth speaker to avoid copying the waveform somewhere.

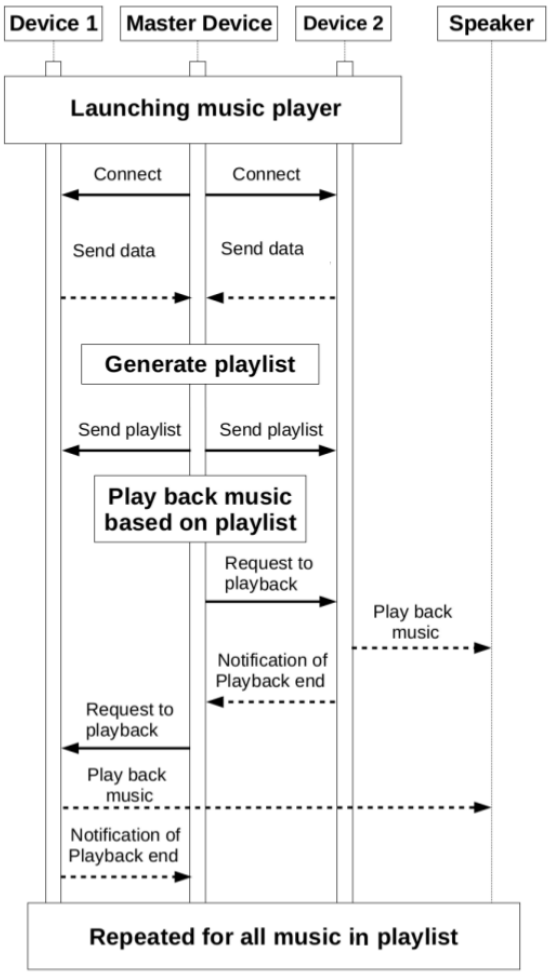


Figure 1. Overview of system flow in the networked playback mode

- Each user's preferences for songs stored on other users' devices is estimated based on the similarity between these songs and the songs stored in his/her device.

The application has two different modes. One is the normal playback mode, in which the user listens to songs in a normal way. This mode provides basically the same functionalities as a typical music player and is used to store the playback history, particularly the number of plays (i.e., how many times the user has listened to each song). The other mode is a networked playback mode, which is the primary mode of this application. Once users gather and launch the application on their devices, the devices start to communicate with one other via Bluetooth, and one of the devices is set to be the master device. After establising aBluetooth connection, the master device collects information about the list of songs stored in each device and the number of plays (how many times each song is played back) from the other devices. Then, the master device generates a playlist and commands the device storing each song in the playlist to play back the song.

Below, we illustrate the procedure of the networked playback mode (Figure 1).

2.1 Launching the Application

Let $\mathcal{U} = \{u_1, \cdots, u_n\}$ and $\mathcal{D} = \{d_1, \cdots, d_n\}$ be a group of users and a set of the users' devices, respectively, where d_1 is the master device. Each device d_i communicates the list of songs $M_{(d_i)} = \{m_{(d_i,1)}, \cdots, m_{(d_i,n(d_i))}\}$ and the number of plays for every song $F(m_{(d_i,k)})$ ($k = 1, \cdots, n(d_i)$) to the master device d_1.

2.2 Calculating the Degree of Preference

For each user u_i, the degree of preference for every song is calculated. Let $W(u_i, m_{(d_j,k)})$ be the degree of preference of user u_i for song $m_{(d_j,k)}$. We formulate the degree of preference based on the following assumptions:

1) If a user listen to a song frequently, his/her preference for that song should be high.

2) If Song A is similar to Song B, which is highly favored, Song A should also highly favored.

Based on the first assumption, we can calculate the degree of a user's preference for songs stored in his/her own device using the number of plays. On the other hand, the degree of preference for songs stored in others' devices cannot be calculated based on the number of plays because such information is not available. Based on the second assumption, we calculate the degree of preference for such songs using that for similar songs owned by oneself.

2.2.1 The Degree of Preference for Owned Songs

The degree of preference for songs stored in a user's own device is calculated based on the number of plays. Here we prepare two different definitions:

$$W(u_i, m_{(d_i,k)}) = 1 - \frac{1}{\{F(m_{(d_i,k)}) + 1\}^{\alpha}} \tag{1}$$

and

$$W(u_i, m_{(d_i,k)}) = \cosh \beta F(m_{(d_i,k)}), \tag{2}$$

where α and β are parameters.

2.2.2 The Degree of Preference for Songs That Are Not Owned

Here, we calculate the degree of preference for songs stored in other users' devices $W(u_i, m_{(d_j,k)})$ ($i \neq j$).

First of all, the similarity of $m_{(d_j,k)}$ to every song stored in d_i is calculated. Musical similarity is a very difficult concept and its calculation is still an open problem. However, various similarity measures have been developed from different points of view (e.g., [2]). Here, we combine two different similarity measures: acoustic similarity and (socially obtained) artist similarity.

To calculate acoustic similarity between two songs, $m_{(d_j,k)}$ and $m_{(d_i,l)}$, a sequence of 20-dimensional mel-frequency cepstral coefficient (MFCC) vectors is calculated from each song by adopting a shift by 160 samples after the waveform is resampled to 16 kHz. The Earth mover's distance between the two sequences, denoted by $D(m_{(d_j,k)}, m_{(d_i,l)})$, is calculated. The similarity $\underset{\text{MFCC}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)})$ is then calculated by using

$$\underset{\text{MFCC}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}) = \frac{1}{1 + D(m_{(d_j,k)}, m_{(d_i,l)})}.$$

The artist similarity between $m_{(d_j,k)}$ and $m_{(d_i,l)}$ is calculated based on the Last.fm API[1] . The Last.fm API has a function, *artist.getSimilar*, which returns up to 100 similar artists to a specified artist, along with similarity values on a scale of 0 to 1. Let $a_{(d_j,k)}$ be the artist of the song $m_{(d_j,k)}$. The list of artists similar to $a_{(d_j,k)}$, denoted by $\mathcal{A}_{(d_j,k)}$, and their similarities $\underset{\text{last.fm}}{\text{sim}}(a_{(d_j,k)}, a')$ ($a' \in \mathcal{A}_{(d_j,k)}$) are obtained using Last.fm. In general, $\underset{\text{last.fm}}{\text{sim}}(\cdot,\cdot)$ is not symmetric. We therefore define the artist similarity as follows:

i) When $a_{(d_j,k)} = a_{(d_i,l)}$,

$$\underset{\text{artist}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}) = 1.0.$$

ii) When $a_{(d_j,k)} \in \mathcal{A}_{(d_i,l)}$ and $a_{(d_i,l)} \in \mathcal{A}_{(d_j,k)}$,

$$\underset{\text{artist}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}) = \frac{1}{2}\left\{\underset{\text{last.fm}}{\text{sim}}(a_{(d_j,k)}, a_{(d_i,l)}) + \underset{\text{last.fm}}{\text{sim}}(a_{(d_i,l)}, a_{(d_j,m)})\right\}$$

iii) When $a_{(d_j,k)} \in \mathcal{A}_{(d_i,l)}$ and $a_{(d_i,l)} \notin \mathcal{A}_{(d_j,k)}$,

$$\underset{\text{artist}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}) = \underset{\text{last.fm}}{\text{sim}}(a_{(d_j,k)}, a_{(d_i,l)})$$

iv) When $a_{(d_j,k)} \notin \mathcal{A}_{(d_i,l)}$ and $a_{(d_i,l)} \in \mathcal{A}_{(d_j,k)}$,

$$\underset{\text{artist}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}) = \underset{\text{last.fm}}{\text{sim}}(a_{(d_i,l)}, a_{(d_j,k)})$$

v) When $a_{(d_j,k)} \notin \mathcal{A}_{(d_i,l)}$ and $a_{(d_i,l)} \notin \mathcal{A}_{(d_j,k)}$,

$$\underset{\text{artist}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}) = \varepsilon,$$

where ε is basically zero but can be set to a very small positive value to take into account the possibility of sparseness in similar artist responses ($\varepsilon = 0.01$ in the current implementation).

The similarity between two songs, denoted by $\text{sim}(m_{(d_j,k)}, m_{(d_i,l)})$, can be calculated as

$$\text{sim}(m_{(d_j,k)}, m_{(d_i,l)}) = \underset{\text{MFCC}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}) \cdot \underset{\text{artist}}{\text{sim}}(m_{(d_j,k)}, m_{(d_i,l)}).$$

Using this similarity measure, the degree of preference can be calculated as follows:

$$W(u_i, m_{(d_j,k)}) = \frac{\sum_l \text{sim}(m_{(d_j,k)}, m_{(d_i,l)}) W(u_i, m_{(d_i,l)})}{\sum_l \text{sim}(m_{(d_j,k)}, m_{(d_i,l)})}.$$

[1] http://www.last.fm/api

2.3 Integration of Degrees of Preference

The degree of every user's preference is integrated as follows:

$$W_{\mathrm{all}}(m_{(d_j,k)}) = \prod_{q=1}^{n} W(u_q, m_{(d_j,k)})$$

2.4 Generating a Playlist

After the integrated degrees of preference $W_{\mathrm{all}}(m_{(d_j,k)})$ for all songs stored in all of the devices are calculated, a necessary number of songs are selected in order of the integrated degree of preference.

2.5 Playing Songs

Let $L = \{m_1, \cdots, m_c\}$ be the list of the selected songs and let $d(m_i)$ be the device storing the song m_i. For each song m_i, the following steps are executed:

1) The master device commands $d(m_i)$ to connect itself to the Bluetooth speaker.

2) The device $d(m_i)$ starts to play back m_i.

3) The master device broadcasts the information (title, artist name, etc.) of the song being played to all devices to alert users about which song is being played.

4) Once the playback ends, the device $d(m_i)$ disconnects from the Bluetooth speaker and sends the master device a message communicating the end of playback.

5) Return to 1) for the next song.

Advanced Audio Distribution Profile (A2DP) is used for the connection between each device and the speaker. For communication between devices, Serial Port Profile (SPP) is used.

3. IMPLEMENTATION AND EXPERIMENTS

3.1 Implementation

We implemented this music player on Android 5.0 smartphones. Screenshots are shown in Figure 2. A demo video of this application is available at `https://www.youtube.com/watch?v=gcOWjkBc_EA`. For roughly one hour, we confirmed that this application sucessfully selected songs, switched the connection to the Bluetooth speaker, and played the selected songs without any troubles.

3.2 Evaluation of the Degree of Preference for Owned Songs

We confirmed the appropriateness of the calculation of the degrees of preference for a user's own songs by checking how these degrees calculated with our method matched the actual level of preference reported by the users.

3.2.1 Dataset

Data of playback histories and preferences were obtained from Last.fm. Using the Last.fm API, we obtained a user profile containing playback histories and evaluations (the "Liked" tag or no tag) for vairous songs. We collected user profiles for 45,745 users, who were specified by choosing one user at random and then traversing "Friend" links recursively. However, many users did not apply the "Liked" tag to any songs, we therefore chose to focus on the 11,074 users who gave "Liked" tags to 20–5000 songs. The total number of the playback histories is 98,504,128.

3.2.2 Results

We analyzed the ratio of songs with the "Liked" tag with respect to the numbers of plays binned by 5 (Figure 3). From this figure, one can see that as the number of plays increases, the ratio of songs with the "Liked" tag increases. Regarding this ratio as the ground truth of the degree of preference, we evaluated the degree of preference calculated with Equations (1) and (2). The results are shown in Figure 4. One can see that using Equation (1) with $\alpha = 0.01$ approximates the ground truth well. For songs with less than 20 times of plays, Equation (1) with $\alpha = 0.10$ and Equation (2) with $\beta = 0.005$ approximate better.

3.3 Evaluation of the Degree of Preference for Songs That Are Not Owned

Next, we confirm the appropriateness of the calculation of the degree of preference for songs not owned by a user.

3.3.1 Dataset

We used the Last.fm Dataset[2] , which consists of playback histories of 1,000 users. Of the songs contained in this database, some songs were regarded as owned songs and other songs were regarded as unowned songs and their playback histories were accordingly hidden. From the playback histories of the songs regared as unowned, the degrees of preference were estimated. Ideally, the estimated degrees should be compared with real favor values (e.g., from questionnaires), but such data were not available from this database. We therefore regarded the degrees of preference calculated from the playback histories as the quasi ground truth. Because this database does not include the waveforms themselves, a 30-second version of every song is downloaded via 7digital API[3] for feature extraction for calculating acoustic similarity.

3.3.2 Procedure

We extracted 118 three-user groups such that the number of songs listened to by three users of every group exceeded 300. For each group, we divided the songs listened to by the group members into three sets; each set was regarded as being owned by each member. We calculated the degrees of preference for unowned songs of each member using the

[2] http://ocelma.net/
[3] https://www.7digital.com/

Figure 2. Screenshot of our Android music player. Left: search of other devices via Bluetooh, Center: playlist display, Right: music playback

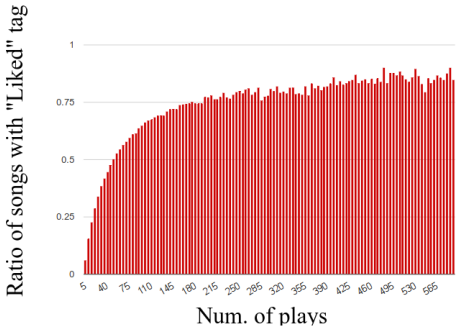
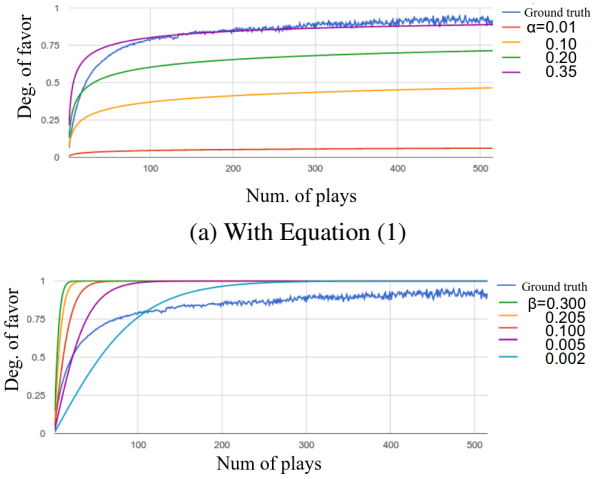


Figure 3. The ratio of songs with the "Liked" tag with respect to the numbers of plays.



(a) With Equation (1)

(b) With Equation (2)

Figure 4. Estimation of degree of preference for owned songs

proposed method. At the same time, the degrees of preference for the same songs were calculated using Equation (1) or (2) and were regarded to be the quasi ground truth.

3.3.3 Results

The correlations between the degrees of preference calculated using our method and the quasi ground truth are listed in Table 1. These data show that the correlation is approximately 0.6 with almost any parameters and therefore that the degree of preference is fairly appropriately estimated. Figure 5 shows the correlations for each user. The correlations were higher than 0.5 for roughly half of the users.

4. CONCLUSION

We have proposed an Android application that makes it possible to seamlesssly enjoy songs that are separately stored on different smartphones or devices. This application has two features. One feature is networked playback. The master device commands other devices to connect to or disconnect from the Bluetooth speaker. Users are accordingly freed from manually switching the device connection. The other feature is music selection. Using each user's playback history and musical similarity, the application estimates the degree of each user's preference even for songs that have not been listened to. Experimental results reveal that the degrees of preference estimated by our method are correlated with the quasi ground truth.

There is still a lot of future work. First, the current

(a) With Equation (1) for owned songs

α	0.01	0.1	0.2	0.3	0.4
Corr.	0.60548	0.60419	0.60217	0.5960	0.59658

(b) With Equation (2) for owned songs

β	0.002	0.005	0.100	0.205	0.300
Corr.	0.56625	0.56708	0.60060	0.60132	0.59658

Table 1. Estimation of degree of preference for unowned songs

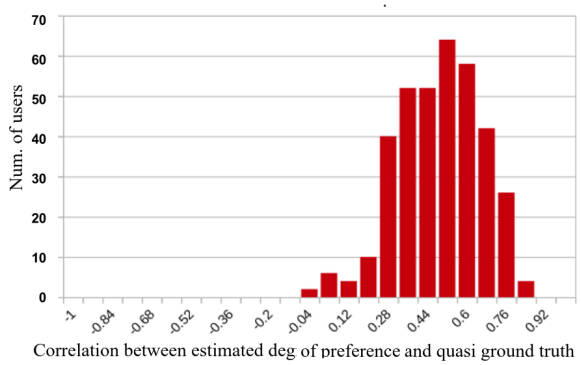


Figure 5. Histogram of per-user correlations between the estimated degree of preference and the quasi ground truth.

playlist generation process—simply placing songs in order of degree of preference—is too naive. We have to improve this process to maintain the users' satisfaction from the beginning to the end of the playlist. We should also conduct usability tests because experiments presented here were focused only on the music selection process. In addition, we plan to distribute this application on Google Play to obtain user feedback for further improvements.

5. ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Numbers 26240025, 26280089, 16K16180, 16H01744, 16KT0136, and 17H00749. We approiate Prof. Tomonobu Ozaki for his fruitful comments.

6. REFERENCES

[1] Implicit personalization of public environments using Bluetooth. In *Proceedings of CHI EA '08 Extended Abstracts on Human Factors in Computing Systems*, pages 3093–3098, 2008.

[2] J.-J. Aucouturier and F. Pachet. Tools and architecture for the evaluation of similarity measures: Case study of timbre similarity. In *Proc. ISMIR*, pages 198–203, 2004.

[3] M. Casey. General sound classification and similarity in mpeg-7. *Organaized Sound*, 6(2), 2002.

[4] A. Crossen and J. Budzik an K. J. Hammond. Flytrap: Intelligent group music recommendation. In *Proceedings of International Conference on Intelligent User Interfaces (IUI '02)*, pages 184–185, 2002.

[5] Pedro Dias and João Magalhães. Music recommendations for groups of users. In *Proceedings of the 2013 ACM international workshop on Immersive media experiences*, pages 21–24. ACM, 2013.

[6] Joseph F. McCarthy and Theodore D. Anagnost. Musicfx: an arbiter of group preferences for computer supported collaborative workouts. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work (CSCW '98)*, pages 363–372, 1998.

[7] Auste Piliponyte, Francesco Ricci, and Julian Koschwitz. Sequential music recommendations for groups by balancing user satisfaction. In *UMAP Workshops*, 2013.

[8] George Popescu and Pearl Pu. What's the best music you have? designing music recommendation for group enjoyment in GroupFun. In *Works-in-Progress of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*, pages 1673–1678, 2012.