Moodful next-track music recommendation based on skipping behaviour

Jacob O'Bryant

25 April 2017

Abstract

Music recommendation systems are needed to help listeners cope with ever-increasing amounts of digital music. We propose a next-track recommender that relies heavily on skipping behaviour. We first implemented a simple system that doesn't respond to short-term preference changes (i.e. changes in mood). We then implemented a new system that does respond to mood changes by using session-based collaborative filtering. Preliminary analysis shows that this implementation is promising. We will conduct more empirical analysis after making a few important enhancements to the algorithm.

1 Introduction

With the ever-increasing amount of digital music available on the internet, information overload has become a significant problem for listeners. To maximize listening pleasure, they must:

1. choose songs to listen to from the set of music they already know and like.
2. continually find new music to add to this set.

Music recommendation systems (MRS) attempt to automate these tasks, but there is much progress to be made. In particular, commercial MRSes often use streaming playback exclusively. Users who already have their own music collections are left with less sophisticated methods to cope with task #1. These methods, such as manually creating playlists, often are either time-consuming or result in suboptimal songs being played.

Our goal is to create an MRS for users who have music collections. The MRS should maximize the user's listening pleasure while minimizing the effort required to use the system. [1] presents our work in evaluating how music recommendation techniques can be used to accomplish this goal. In short, we propose a moodful next-track recommender based on skipping behaviour. By "moodful," we mean that the system is aware that users like different music at different times and that the system automatically detects and responds to these mood shifts. Skipping behaviour is the only user input. The system automatically chooses the next track to play whenever the current track finishes, and it learns the user's preferences based on whether they completed the song or skipped it before completion.

This paper describes our current efforts in implementing Smart Shuffle Player, a prototype of this proposed system. It is a work in progress. In future work, the data collected from this system could be used to handle task #2.

2 Related Work

[1] gives an in-depth comparison of the proposed system with several academic and commercial MRSes. Table 1 provides a summary of the main differences.

3 Methods

Smart Shuffle Player is a modification of the open-source Android app "Vanilla Music Player." We replaced the code that runs when the current song finishes so we can specify which song to play next.

For simplicity, we first implemented a "moodless" algorithm. We implemented the moodful algorithm after that was done successfully. Both algorithms attempt to learn a probability distribution for the user's library, i.e. each song is assigned a probability that it would be acceptable to the user if played next. The algorithms then make a weighted random choice for which song to play based on these probabilities.

Both algorithms also use the same data input. Whenever a song is completed, the app records the song metadata (title, album and artist), whether or not the song was skipped or completed, and the current timestamp. Given this data, the algorithms return the metadata of a song to be played next.

3.1 Moodless implementation

The moodless algorithm assumes that the user has only a single mood, i.e. there is one true probability distribution that does not change significantly in the short-term. It also assumes that tracks from the same album or artist tend to be similar.

We used the following formula to calculate for each song the probability P that the user will not skip the song if we play it:

$$\begin{aligned} P = & R_{\text{song}} C_{\text{song}} + (1 - C_{\text{song}}) \times \\ & (R_{\text{album}} C_{\text{album}} + (1 - C_{\text{album}}) \times \\ & (R_{\text{artist}} C_{\text{artist}} + (1 - C_{\text{artist}}) r)) \end{aligned}$$

where R is the ratio of times skipped to times played, C is a confidence level from 0 to 1 and r is a constant default ratio. C is given by

$$C(n) = \frac{n}{n + a}$$

where n is the number of times the song has been played and a is an arbitrary constant. a can be tuned to adjust how rapidly the confidence level grows.

We allowed the user to manually specify different moods. The app stored a discrete probability distribution for each mood. This allowed us to preserve the assumption that the probability distribution doesn't change in the short-term.

To measure the algorithm's performance, we modified the app so that occasionally it would randomly pick the next song to play without using the moodless algorithm's probability distribution. We compared the user's overall skip ratios in response to the smart shuffle algorithm and this pure shuffling.

In preliminary tests done with a single user, we found a statistically significant improvement when using the smart shuffle algorithm. This encouraged us to move on to implementing the moodful algorithm rather than conducting further data analysis.

3.2 Moodful Implementation

We originally considered an approach that maintained the moodless algorithm's discrete probability distributions but dynamically detected transitions in the user's mood. However, we decided instead to create a completely new algorithm that uses session-based collaborative filtering. This approach groups

	Recommender type	Main data source	Moodful
Smart Shuffle Player	next-track with local files	skipping behaviour	yes
Pandora	next-track with streaming playback	content-based model (Music Genome Project)	no
Last.fm	no playback; recommendations delivered via user profiles on the website	Scrobbling (many music players can report to last.fm which songs the user plays, but this does not include songs that the user skips)	no
Pampalk's system[2]	next-track with local files	content-based model	yes
NextOne Player[3]	next-track with local files	content-based model	yes

Table 1: Comparison of MRSes

the user's listening events (skips and completions) into sessions. It assumes that the user's mood does not change within a single session.

The algorithm has the following basic steps:

1. Cluster the events into sessions based on timestamp.

2. Assign each previous session a score based on similarity to the current session.

3. Take the k most similar sessions and probabilistically choose a song that 1) was completed in one of the sessions, and 2) hasn't been played yet in the current session.

4. If there are no songs that meet those criteria, randomly pick a song from the library.

We haven't done any quantitative evaluation of the moodful algorithm yet because it is still in early development (see Future Work). Qualitatively speaking, the algorithm quickly learns and responds to the user's skipping behaviour. However, the algorithm tends to play the same songs from session to session without exploring the user's library.

4 Future Work

Wang states, "A successful recommender system must balance the needs to explore user preferences and to exploit this information for recommendation."[4] The problem of exploration vs. exploitation has been widely studied in machine learning research, but it has not been widely incorporated into music recommendation research. [4] and [5] demonstrate empirically that exploration can improve the performance of both collaborative and content-based systems, respectively.

An element of exploration needs to be added to the moodful algorithm. This could be done with the help of a hybrid approach

that uses a content-based model to guess song similarity when there is insufficient skipping data.

Additionally, the system should give less weight to songs that have been played recently. This concept of "freshness" has been thoroughly explored in [3].

The app currently uses a simple memory-based collaborative filtering approach. Although scaling is not a concern right now, the quality of the recommendations would likely be improved by switching to a model-based collaborative filtering approach. For instance, this would take more advantage of shorter sessions that normally wouldn't be included in the k most similar sessions.

5 Conclusion

We have made significant progress in figuring out what direction to go with this project, and early prototypes are promising. The proposed moodful next-track recommender appears to be an effective solution. After it is further developed according to the ideas included in Future Work, we will recruit more users and evaluate the system empirically. Once we gather enough data, we can experiment with recommending new songs to users.

References

[1] J. O'Bryant, "A survey of music recommendation and possible improvements," 2017. [Online]. Available: http://jacobobryant.com/about/mrs.pdf (visited on 04/25/2017).

[2] E. Pampalk, T. Pohle, and G. Widmer, "Dynamic playlist generation based on skipping behavior.," in *ISMIR*, vol. 5, 2005, pp. 634–637. [Online]. Available: http://cis.ofai.at/~elias.pampalk/publications/pam_ismir05b.pdf (visited on 03/05/2017).

[3] Y. Hu and M. Ogihara, "NextOne player: A music recommendation system based on user behavior.," in *ISMIR*, 2011, pp. 103–108. [Online]. Available: http://www.academia.edu/download/38297540/PS1-11.pdf (visited on 03/05/2017).

[4] X. Wang, Y. Wang, D. Hsu, and Y. Wang, "Exploration in interactive personalized music recommendation: A reinforcement learning approach," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 11, no. 1, pp. 1–22, Sep. 4, 2014, ISSN: 15516857. DOI: 10.1145/2623372. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2665935.2623372 (visited on 03/08/2017).

[5] Z. Xing, X. Wang, and Y. Wang, "Enhancing collaborative filtering music recommendation by balancing exploration and exploitation.," in *ISMIR*, 2014, pp. 445–450. [Online]. Available: http://www.terasoft.com.tw/conf/ismir2014/proceedings/T081_140_Paper.pdf (visited on 04/05/2017).