A survey of music recommendation and possible improvements

Jacob O'Bryant

07 April 2017

Abstract

With the prevalence of the internet, mobile devices and commercial streaming music services, the amount of digital music available is greater than ever. Sorting through all this music is an extremely time-consuming task. Music recommendation systems search through this music automatically and suggest new songs to users. Music recommendation systems have been developed in commercial and academic settings, but more research is needed. The perfect system would handle all the user's listening needs while requiring only minimal user input. To this end, I have reviewed 20 articles within the field of music recommendation with the goal of finding how the field can be improved. I present a survey of music recommendation, including an explanation of collaborative and content-based filtering with their respective strengths and weaknesses. I propose a novel next-track recommendation system that incorporates techniques advocated by the literature. The system relies heavily on user skipping behavior to drive both a content-based and a collaborative approach. It uses active learning to balance the needs of exploration vs. exploitation in playing music for the user.

1 Introduction

With the ever-increasing amount of digital music available on the internet, the task of finding new music has become very difficult. Music recommendation systems help deal with this information overload by automatically recommending new music to listeners.

Music recommendation systems have been developed both in industry and academia. Industrial systems such as Pandora and Spotify have achieved commercial success, but they have not reached perfection. Song observes that from an academic perspective, "the development of music recommender systems is still at a very early stage."[1] The challenge of music recommendation is to create a system that can continually find appealing new music while minimizing the amount of user effort involved.

This paper presents a survey of music recommendation with the goal of finding how it can be improved. First I describe the general approaches to music recommendation, including their strengths and weaknesses. I then discuss techniques that are commonly used to counteract these weaknesses. I propose a new recommendation system that incorporates these techniques. Finally, I analyze existing recommendation systems and show how the new system would be a novel contribution.

2 Methodology

I first did a broad search on SCOPUS and Google Scholar for music recommendation and music information retrieval. I selected articles that provide an overview of the field, especially those that focus on collaborative filtering. I also selected articles that present experimental data from implemented recommendation systems. One such system was given in [2]. I searched for articles that cite this one. I selected articles that discuss skipping behavior and other methods that could be used to improve recommendation.

3 General approaches

[1] provides a broad survey of music recommendation techniques. The two main approaches are collaborative filtering and content-based filtering. [3] provides a more in-depth explanation of collaborative filtering. The following discussion draws heavily on these sources.

3.1 Collaborative filtering

Collaborative systems generate recommendations by comparing ratings of items between different users. Collaborative systems do not attempt to model the actual item being recommended; they only analyze users' responses to those items. Collaborative systems are built on the assumption that users who rate items similarly in the past will continue to rate them similarly in the future. Last.fm is an example of a prominent collaborative system.

Given a set of m items and a set of n users, a collaborative system can construct an $m \times n$ matrix of user-item ratings (this matrix will often be sparse since most users have exposure to only a few items). Using matrix factorization techniques, the system can identify which users have rated items similarly. Given two similar users i and j, recommendations for i can be chosen from the positively-rated items of j and vice-versa. This is called the k Nearest Neighbor algorithm.

Ratings can be either explicit or implicit. Examples of explicit ratings include one-to-five star ratings on Amazon.com or thumbs-up and thumbs-down feedback on Pandora. These ratings require explicit input from the user. Even if the user doesn't give an explicit rating for an item, an implicit rating can often be inferred from user behaviour. [4] shows that a variety of data sources can effectively reveal implicit ratings. Play counts can be used to infer an implicit rating; a song that has been played many times would be given a high implicit rating. Playlists are a particularly prevalent source of implicit ratings.[5, 6]

3.1.1 Challenges

Data sparsity is one of the biggest problems for collaborative systems.[1] The system needs a large amount of data about the users and the potential items to generate effective recommendations. If this data is not available, the system will be ineffective. A specific instance of the data sparsity problem is the cold-start problem. This occurs when a new user or item is introduced to the system. The system can't give effective recommendations to a new user because the user hasn't rated any items yet. A new item cannot be effectively recommended to existing users because it doesn't have any ratings yet.

Another problem for collaborative systems is scalability.[3] When the system has data for a large number of items and users, the recommendation algorithm can take a long time to process the data. This is especially a problem for systems that must generate recommendations on the fly such as a mobile music player.

Human effort is a third challenge.[1] The

more effort it takes for users to rate items, the fewer ratings they will give. This is especially applicable for explicit ratings. High human effort contributes to the data sparsity problem.

An additional challenge is popularity bias. Items with more ratings will tend to be recommended more frequently than items with fewer ratings, regardless of the ratings' values.[1] Less well-known items are said to be in the "long tail" of the popularity curve. Recommendation systems should be able to effectively recommend items from the long tail.

3.2 Content-based filtering

Instead of comparing user ratings, content-based systems construct models of the items being recommended and compare them to the preference model of the current user.

Item models for songs can be generated in a number of ways. [7] presents a content-based system that analyzes audio files to determine song features such as loudness, tempo and timbre. They find that this algorithm helps to mitigate the cold-start problem, i.e. when little data is already known about the song being recommended. It is also common for music experts to analyze songs and apply appropriate tags by hand. The Music Genome Project contains a large database of song information of this kind. This database is the core of Pandora's recommendation system.

The user's preference model doesn't necessarily contain any information about specific songs that the user does or does not like. Instead, it records the user's response to each of the components in the item model. For instance, the item model may include the gender of the vocalist. If the user consistently has a positive response to songs with female vocals, the user preference model will likely indicate a positive bias towards female vocals.

3.2.1 Challenges

The major problem with content-based filtering is that it relies on the correctness of the item model.[1] The system is thus limited by what it understands about the music. Regardless of how much user input is collected, the system is unable to overcome limitations in the item models. Schedl describes this problem as the "glass-ceiling effect" of music recommendation.[8]

For example, the item model may not take into account important differences between otherwise similar songs. The system may not know the difference between a hard rock song with melodic lyrics and a hard rock song with screamed lyrics. It might then repeatedly recommend songs with screamed lyrics to users who only like melodic hard rock.

4 Common design techniques

So far I have laid a foundation for how music recommendation works. Now I describe methods that are used to counteract the challenges of the two recommendation approaches.

4.1 Hybrid systems

Content-based filtering is like a car with high acceleration but low top speed. It can give effective recommendations quickly because it doesn't need a lot of users, but the effectiveness of those recommendations won't improve after the system does gain many users. Collaborative filtering is the opposite; it has high top speed but low acceleration.

As such, the two approaches can be combined into a hybrid system that draws on the strengths of each.[1] Collaborative and content-based filtering can be combined in

many ways. One method is to use collaborative filtering as the default system while content-based filtering is used when the collaborative system lacks sufficient data about a particular item. Methods like these help to mitigate the inherent challenges of each approach, such as data sparsity. Experimental studies show that hybrid systems outperform systems that rely on only one recommendation approach.[9, 10]

4.2 Active learning

[11] The recommendation problem can phrased as "given a user and a set of items, which item would receive the most positive response from the user?" After the song is recommended, the system can improve based on the user's response. As such, the recommendation problem could be rephrased as "given a user and a set of items, which item would teach the system the most about the user's preferences if we recommended it?" Active learning is the process of giving recommendations that will help the system to learn as much as possible. This technique helps to quickly overcome data sparsity.

For example, suppose a user has given a positive response to many songs from artist A but has never heard a song from artist B. A new, unheard song from artist A would likely receive a positive response, but a response to a song from artist B would tell the system more about the user's preferences.

Both formulations of the recommendation problem are important. Wang states, "A successful recommender system must balance the needs to explore user preferences and to exploit this information for recommendation."[12] 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. [12] and [11] demonstrate empirically that active learning can improve the performance of both collaborative and content-based systems, respectively.

4.3 Listening history

Many different data sources can be used to infer users' music preferences. The richer the data source, the better the recommendation system will be. User listening histories are a particularly useful data source that can be easily collected from digital music players.[13] This data can show overall which songs the user prefers the most; for example, a song that has been listened to 100 times is likely more favored than a song that has been listened to only 10 times. However, comprehensive listening histories can reveal much more than this.

[14] demonstrates how deeper inferences about music preferences can be made by partitioning the listening histories into sessions. These sessions can indicate the similarities between songs in the user's collection. Songs are more likely to be similar if the user listens to them in succession. [15] demonstrates that temporal aspects from the history can also improve recommendation. They present a model that examines which songs the user listens to based on different temporal factors like time of day and day of week. [16] shows that temporal context can be combined with session-based collaborative filtering for further improvement. Finally, [17] uses listening histories to differentiate between the user's short-term and long-term preferences.

4.4 User-centric design

Schedl demonstrates an important design flaw in many music recommendation studies.[18] When measuring the performance of new systems, many studies compare new results against existing datasets. [19] compares music recommendation systems using a common evaluation technique. Given a dataset of

human-created playlists, the last song from each playlist is removed. Then the dataset is given to each recommendation system, and the systems pick the next song in the playlist. Performance is based on how often the system chooses the same song as the original creator of the playlist. The authors compare four academic recommendation systems and one commercial playlisting service. They find that the commercial service had the least accuracy.

However, the commercial service had been tested and optimized on actual users for many years. Although it didn't perform well in this academic comparison, it is likely that it would have performed better in a test that required actual humans to rate each system. Schedl argues that design and evaluation of recommendation systems must include real users, even though this increases the cost and complexity of experiments.

5 Suggested system

Commercial recommendation systems like Pandora and Spotify have focused on streaming music instead of playing music that the user already owns. This avoids the problem of having to manage a large music collection; however, many people already have music collections. These collections provide a rich data source that would be useful for recommendation. Our system focuses first on these users.

A new music player application could at first use content-based filtering and active learning to explore the user's music collection. Instead of forcing the user to create playlists or otherwise select which songs to play, the app would choose automatically the next track to play. This setup is known as next-track recommendation. The user's skipping behavior would be recorded and used to construct a complete listening history. Since skipping behaviour includes both songs the user listens to and songs the user skips, the history would include songs that the user didn't want to listen to at a certain time (i.e. failed recommendations). The skipping behavior also gives a user-centric way to evaluate the system's performance.

After the app gains enough users, it could gradually switch over to a collaborative approach in order to recommend new songs. These recommendations could be streamed to the music player to minimize human effort, and feedback would still be collected through skipping behaviour. This would be a hybrid system in two different ways: the system combines collaborative and content-based filtering, and it combines streaming playback with local playback.

6 Related systems

The individual parts of our new system have already been studied, but I was not able to find any systems that use this particular combination of those ideas. I now analyze several similar recommendation systems to contrast them with the proposed system.

6.1 Academic systems

Pampalk presented one of the first music players that uses skipping behavior for next-track recommendation.[2] The player uses a simple content-based model to represent the similarity between songs in the user's library. The user's skipping behavior determines their preference model. Experimental results showed that the number of skips was dramatically reduced when using this algorithm as opposed to other baseline algorithms such as pure shuffling. This system relies on skipping behaviour like our suggested system. However, it does not incorporate collaborative filtering or active learning.

NextOne player is a more recent system

that also uses a content-based model combined with skipping behavior.[20] It uses the listening history to calculate a "freshness" value for each song. This allows the player to give more weight to songs that haven't been played recently. Although the app uses the listening history, it doesn't do any session-based analysis. It also doesn't incorporate active learning or collaborative filtering; however, the authors do recommend collaborative filtering as a subject for future work.

6.2 Commercial systems

Last.fm is a widely known collaborative system. One if its data sources is called scrobbling. Scrobbling is a feature of many music players that allows the user's listening history to be uploaded to last.fm. This is somewhat similar to our proposed system because it is a collaborative system based on listening history; however, there are critical differences. Scrobbling does not do any next-track recommendation, it only lists the songs you play on your own. As such, the last.fm system does not benefit from active learning. It also suffers from the human effort problem because it is not integrated seamlessly with a music player.

As mentioned before, Pandora is a content-based system. Pandora mainly suffers from the inherent glass-ceiling effect of content-based filtering. It also cannot take advantage of users' existing collections since it only streams music.

7 Future Research

I am currently developing the suggested system; it is available for download as an Android app called Smart Shuffle Player. It uses skipping behavior to power a simple content-based model. I am currently researching how to implement session-based collaborative filtering and active learning. This will be enough for the player to do effective next-track recommendation with only the user's current library. Additional research will be needed to connect the app with a centralized collaborative system in order to recommend new songs from other users.

8 Conclusion

I have presented a survey of the state of the art in music recommendation with a focus on collaborative and content-based filtering. I have found that a hybrid recommender system built on user skipping behavior could be a useful next step in recommendation system design. This system has the potential to provide users with a music experience that seamlessly integrates local playback with streaming playback for effective recommendations while requiring minimal human effort.

References

[1] Y. Song, S. Dixon, and M. Pearce, "A survey of music recommendation systems and future perspectives," in *9th International Symposium on Computer Music Modeling and Retrieval*, 2012. [Online]. Available: `https://www.researchgate.net/profile/Yading_Song/publication/277714802_A_Survey_of_Music_Recommendation_Systems_and_Future_Perspectives/links/5571726608aef8e8dc633517.pdf` (visited on 03/05/2017).

[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] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in Artificial Intelligence*, vol. 2009, e421425, Oct. 27, 2009, ISSN: 1687-7470. DOI: 10.1155/2009/421425. [Online]. Available: https://www.hindawi.com/journals/aai/2009/421425/abs/ (visited on 03/05/2017).

[4] B. Yang, S. Lee, S. Park, and S.-g. Lee, "Exploiting various implicit feedback for collaborative filtering," in *Proceedings of the 21st International Conference on World Wide Web*, ACM, 2012, pp. 639–640. [Online]. Available: http://dl.acm.org/citation.cfm?id=2188166 (visited on 03/28/2017).

[5] B. Fields, "Contextualize your listening: The playlist as recommendation engine," PhD thesis, Department of Computing Goldsmiths, University of London, 2011. [Online]. Available: https://www.researchgate.net/profile/Benjamin_Fields2/publication/266271406_Contextualize_Your_Listening_The_Playlist_as_Recommendation_Engine/links/561e2d5108aef097132b32bc.pdf (visited on 03/05/2017).

[6] F. Maillet, D. Eck, G. Desjardins, P. Lamere, and others, "Steerable playlist generation by learning song similarity from radio station playlists.," in *ISMIR*, 2009, pp. 345–350. [Online]. Available: http://www.ismir2009.ismir.net/proceedings/OS4-2.pdf (visited on 03/05/2017).

[7] S.-Y. Chou, Y.-H. Yang, J.-S. R. Jang, and Y.-C. Lin, "Addressing cold start for next-song recommendation," ACM Press, 2016, pp. 115–118, ISBN: 978-1-4503-4035-9. DOI: 10.1145/2959100.2959156. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2959100.2959156 (visited on 04/05/2017).

[8] M. Schedl, E. Gmez, J. Urbano, and Now Publishers, *Music information retrieval: recent developments and applications*. 2014, OCLC: 905837683, ISBN: 978-1-60198-807-2. [Online]. Available: http://dx.doi.org/10.1561/1500000042 (visited on 04/04/2017).

[9] Bo Shao, M. Ogihara, Dingding Wang, and Tao Li, "Music recommendation based on acoustic features and user access patterns," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 8, pp. 1602–1611, Nov. 2009, ISSN: 1558-7916. DOI: 10.1109/TASL.2009.2020893. [Online]. Available: http://ieeexplore.ieee.org/document/5230332/ (visited on 03/13/2017).

[10] M. Schedl, "Ameliorating music recommendation: Integrating music content, music context, and user context for improved music retrieval and recommendation," in *Proceedings of International Conference on Advances in Mobile Computing & Multimedia*, ACM, 2013, p. 3. [Online]. Available: http://dl.acm.org/citation.cfm?id=2536856 (visited on 03/05/2017).

[11] 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).

[12] X. Wang, Y. Wang, D. Hsu, and Y. Wang, "Exploration in interactive personalized music recommendation: A re-

inforcement 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).

[13] D. Baur and A. Butz, "Pulling strings from a tangle: Visualizing a personal music listening history," in *Proceedings of the 14th international conference on Intelligent user interfaces*, ACM, 2009, pp. 439–444. [Online]. Available: http://dl.acm.org/citation.cfm?id=1502715 (visited on 03/05/2017).

[14] S. E. Park, S. Lee, and S.-g. Lee, "Session-based collaborative filtering for predicting the next song," in *Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on*, IEEE, 2011, pp. 353–358. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5954341/ (visited on 03/28/2017).

[15] C. H. Park and M. Kahng, "Temporal dynamics in music listening behavior: A case study of online music service," IEEE, Aug. 2010, pp. 573–578, ISBN: 978-1-4244-8198-9. DOI: 10.1109/ICIS.2010.142. [Online]. Available: http://ieeexplore.ieee.org/document/5591002/ (visited on 03/05/2017).

[16] R. Dias and M. J. Fonseca, "Improving music recommendation in session-based collaborative filtering by using temporal context," IEEE, Nov. 2013, pp. 783–788, ISBN: 978-1-4799-2972-6 978-1-4799-2971-9. DOI: 10.1109/ICTAI.2013.120. [Online]. Available: http://ieeexplore.ieee.org/document/6735331/ (visited on 03/08/2017).

[17] I. Kamehkhosh, D. Jannach, and L. Lerche, "Personalized next-track music recommendation with multi-dimensional long-term preference signals," [Online]. Available: http://ceur-ws.org/Vol-1618/IFUP_paper_3.pdf (visited on 03/08/2017).

[18] M. Schedl, A. Flexer, and J. Urbano, "The neglected user in music information retrieval research," *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 523–539, Dec. 2013, ISSN: 0925-9902, 1573-7675. DOI: 10.1007/s10844-013-0247-6. [Online]. Available: http://link.springer.com/10.1007/s10844-013-0247-6 (visited on 03/05/2017).

[19] D. Jannach, I. Kamehkhosh, and G. Bonnin, "Biases in automated music playlist generation: A comparison of next-track recommending techniques," ACM Press, 2016, pp. 281–285, ISBN: 978-1-4503-4368-8. DOI: 10.1145/2930238.2930283. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2930238.2930283 (visited on 04/05/2017).

[20] 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).