Music Recommendation Based on Artist Novelty and Similarity

Ning Lin, Ping-Chia Tsai, Yu-An Chen #, Homer H. Chen

Department of Electrical Engineering, National Taiwan University

Graduate Institute of Communication Engineering, National Taiwan University

1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan

{b99901006, b99901157, b96901042, homer}@ntu.edu.tw

***Abstract*—Most existing systems recommend songs to the user based on the popularity of songs and singers. However, the system proposed in this paper is driven by an emerging and somewhat different need in the music industry—promoting new talents. The system recommends songs based on the novelty of singers (or artists) and their similarity to the user's favorite artists. Novel artists whose popularity is on the rise have a higher priority to be recommended. Specifically, given a user's favorite artists, the system first determines the candidate artists based on their similarity with the favorite artists and then selects those who have a higher novelty score than the favorite artists. Then, the system outputs a playlist composed of the most popular songs of the selected artists. The proposed system can be integrated into most existing systems. Its performance is evaluated using the Spotify Radio Recommender as a reference and a pool of 100 subjects recruited on campus. Experimental results show that our system achieves a high novelty score and a competitive user-preference score.**

I. INTRODUCTION

It is relatively easy to access music thanks to the advance of internet technology. Yet finding one's favorite music from millions of songs is often not as easy as one would expect. From the user's perspective, it is desirable to automatically receive songs that match the user's expectation or taste from online music providers without manually inputting any keyword. Among all approaches to music recommendation, the one guided by artist similarity appears to be plausible because it ensures that the performing style of the recommended singers is to the liking of the user. Indeed, such an approach is adopted by many services [1], [2]. For example, given the favorite artists input by the user, the Spotify Radio identifies additional artists similar to the favorite artists and recommends their songs as well. As another example, Last.fm identifies the similarity between artists based on the users' listening history.

However, these systems tend to recommend popular artists and often suffer from two drawbacks. First, the recommended songs may have already been listened to by the user. It would be more refreshing to provide the user with unexpected and fortuitous music experience [3]-[5]. Second, hidden jewels in the music community, such as indie singers, are rarely recommended to the users and may be left undiscovered [6]. In this paper, we propose a system to address these drawbacks by considering both artist similarity and artist novelty.

The contributions of this paper are twofold. First, our approach recommends novel music effectively with high user satisfaction. It works for a wide range of music from pop, electronic, metal, jazz, rock, hip hop, country, hardcore, to vocal music. Second, the system is robust and capable of discovering new artists efficiently.

The rest of this paper is organized as follows. In Section 2, we review three major approaches to music recommendation. Section 3 describes the proposed recommendation system in details. The performance evaluation of the recommendation system is reported in Section 4. Finally, Section 5 concludes the paper and discusses future work.

2014 IEEE 16th International Workshop on Multimedia Signal Processing (MMSP), Sep. 22-24, 2014, Jakarta, Indonesia.
978-1-4799-5896-2/14/$31.00 ©2014 IEEE.

II. RELATED WORK

Over the past few years, three major types of recommendation techniques have been developed: content-based, collaborative filtering (CF), and hybrid (a combination of the former two). This section gives a review of these techniques.

A. Content-Based Filtering

Assuming that items (i.e., song or artist) with similar properties are equally attractive to the user, content-based methods model user preference by learning the association between user's rating of the items and the properties of the items with various machine learning methods, such as decision trees [7], neural networks [8], or Bayesian networks [9]. The acoustic properties of songs, such as rhythm, tempo, frequency spectrum, genre, etc., are often used as the features in the learning process.

A major advantage of content-based filtering is that it is free of the new item problem (which refers to the inability of a system to recommend new items that have not been rated by any user [10]) so that well-known and less-known artists have equal chance to be recommended. However, a major drawback of the content-based filtering approach is that it considers that each user is independent of the others [11]. Because it ignores the social nature of human beings, it suffers from over-specialization and is only able to recommend items similar to those in the user's listening history.

B. Collaborative Filtering

People's music tastes may influence one another, especially within a social community. Because the music taste of a user is an important piece of information for music recommendation, collaborative filtering (CF) predicts a user's preference from the music listening history (or user profile) of the other users' tastes [12]-[14]. CF approaches can be memory- or model-based. Memory-based recommenders compute the similarity between users by correlation or other similarity measures [15], [16], and the resulting similarity is used to predict the preference of a user. On the other hand, model-based recommenders use the user profiles to train a model by data mining or machine learning techniques, such as Bayesian network [13], latent semantic modelling [17], [18], and Markov decision modelling. Then the resulting model is used to predict the preference of a user.

However, CF has the following four drawbacks. First, it is not as scalable as content-based filtering. When the total amount of user data is large, CF becomes inefficient. When it is small, data sparsity problem is unavoidable [19]. Second, cold start often occurs to a new user because there is not enough rating record of the user. Third, newly released albums or music have little chance to be recommended. Finally, users with unusual tastes receive poor predictions since they share little similarity with others—this is the so-called gray sheep problem [20].

Despite of the drawbacks, the greatest strength of CF is that it is completely independent of the music content. Unlike content-based filtering, CF relies merely on user ratings and is able to recommend items without any additional data.

C. Hybrid Recommender System

Both content-based filtering and collaborative filtering have pros and cons [21]. Hybrid approaches gain better performance by combining the two in different ways: 1) combining the scores of CB and CF together to produce a single recommendation, 2) presenting recommendations from different recommenders at the same time, and 3) developing a unified model through feature fusion or model cascading.

III. System Overview

The proposed music recommendation system is designed to meet the following three requirements:

- The music recommendation system should be able to deal with a relatively small list of favorite singers given by the user.
- The artists of the recommended songs should be new to the user.
- The user acceptance rate should be reasonably high.

Recommending music unheard by a user helps the user to discover new artists and enables the music industry to promote uprising talents who are not yet popular. However, the condition that only the information of favorite artists is available presents a challenge in the system design. If additional user information such as listening history is available, the task becomes relatively easier because we may analyze the listening history of the user to find out the user behavior.

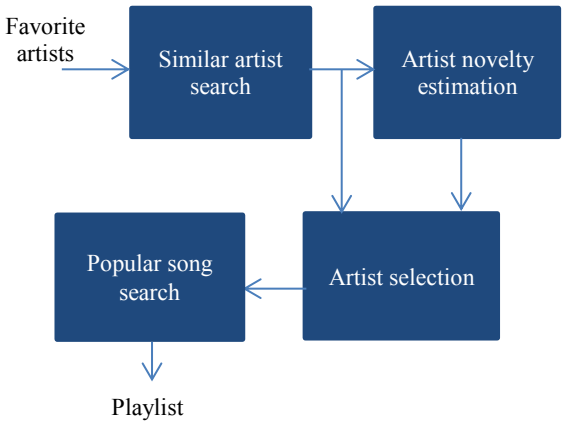

Fig. 1. System flow

The system flow is depicted in Fig. 1. Each component of the system is discussed in detail in this section.

A. Terminology

The following two labels are used for artist classification:

- *Like/dislike*: This label shows whether an artist is liked/disliked by a user.
- *New/known*: This label indicates whether an artist is new to a user or not.

These two labels describe the relationship between artists and users. When an artist is labelled new, the like/dislike label is ignored. After an artist becomes known to the user, the like/dislike label is activated.

The following two artist attributes are defined:

- *Similarity*: The more characteristics (such as music genre, timbre, tag, and era) two artists share, the more similar the two artists are.
- *Popularity*: The degree popularity of an artist.

Note that a popular artist is more likely to be well known to the public, and an artist who is similar to a favorite (liked) artist of a user has a higher chance to be liked by the user. *New* and *popularity* are used together to define artist novelty. *Like* and *similarity* are used to predict the level of user's liking of an artist.

B. Similar Artist Search

As mentioned, artists who are similar to the favorite artists of a user are more likely to be liked by the user. Given a favorite artist a of a user and another artist b, the probability that the user would like artist b is related to the similarity between a and b and can be expressed as

$$P(like|\, a, b) \propto sim(a, b)\,, \tag{1}$$

where $sim(\cdot,\cdot)$ denotes similarity. Therefore, the level of a user's liking of an artist can be obtained from the similarity without computing the conditional probability. The similarity between artists can be computed based on information such as artist timbre, style, and cultural background. Some music service companies have provisioning for artist similarity measurement. For example, Last.fm computes such similarity

data every few minutes by combining content-based information (i.e., tags) and collaborative information. The data can be access through an API [22]. Due to its efficiency, we use this API in this paper to access the artist similarity information.

Our system uses the list of favorite artists a_1, a_2, ... a_n of a user as input,

$$A = [a_1, a_2, \dots\dots, a_n]. \quad (2)$$

The list can be obtained from the user's social website. For example, a user's preference information of an artist can be obtained from the artist's fan page on Facebook [23].

For every artist a_i in A, we search for the top N similar artists denoted by $S(a_i)$ according to the similarity score $sim(a_i)$:

$$S(a_i) = [b_{i,1}, b_{i,2}, \dots\dots, b_{i,N}], \quad (3)$$

$$sim(a_i) = [sim(a_i, b_{i,1}), \dots\dots, sim(a_i, b_{i,N})], \quad (4)$$

where $b_{i,j}$ is the jth similar artist of a_i.

C. Artist Popularity Estimation

Artist novelty is the basis of recommendation in our system, which recommends artists new to the user and thereby brings fresh music experience to the user. As mentioned, a popular artist is likely to be labelled known for the user. Given a user u and a candidate artist $b_{i,j}$, the conditional probability that the artist can be labelled known is proportional to the popularity of the artist in the eyes of the user. That is,

$$P(known|u, b_{i,j}) \propto pop(b_{i,j}|u), \quad (5)$$

where $pop(b_{i,j}|u)$ denotes the popularity of $b_{i,j}$ viewed by u. Therefore, we determine the new/known label of an artist with respect to a user from the popularity of the artist without computing the conditional probability. Note that the conditional probability that the artist can be labelled new can be considered as the degree of novelty of the artist with respect to the user. Therefore, we have

$$\begin{aligned} Novelty(b_{i,j}|u) &= P(new|u, b_{i,j}) \\ &= 1 - P(known|u, b_{i,j}) \\ &= 1 - pop(b_{i,j}|u) \end{aligned} \quad (6)$$

where $Novelty(\cdot)$ denotes novelty. In our system, $pop(b_{i,j}|u)$ is obtained by normalizing the popularity of $b_{i,j}$ by the popularity of a_i. That is,

$$pop(b_{i,j}|u) = \frac{log\ pop(b_{i,j})}{log\ pop(a_i)}, \quad (7)$$

where $pop(\cdot)$, which denotes popularity, is obtained by the total count of responses to a Google search with the name of artist as the key word.

Note that only candidate artists with popularity in the eyes of the user smaller than a threshold are kept in the candidate list. In our implementation, the threshold is set to 1. Therefore, only candidate artists who are less famous than the corresponding favorite artist are considered, and any artist whose popularity is higher than the favorite artist is labelled known. Also note that the similarity detection and the popularity estimation are performed sequentially so that the computation required for popularity estimation is limited to only similar artists. If these two operations are performed simultaneously, more computation is required although the same result is obtained.

D. Artist Selection and Popular Song Search

Similarity of a candidate artist with respect to a favorite artist of a user reflects the preference level of the user for the candidate artist. Thus, the recommendation score of a candidate artist is proportional to its similarity to the favorite artist. Furthermore, as the recommendation is also based on novelty, the final recommendation score for a candidate artist is obtained by multiplying the two factors together. That is,

$$Score(b_{i,j}) = Sim(a_i, b_{i,j}) \times Novelty(b_{i,j}), \quad (8)$$

where $Score(\cdot)$ denotes the recommendation score of the candidate artist given a_i is a favorite artist of the user. Among the candidate artists similar to a_i, the one with the highest recommendation score is selected, and the most popular song of this artist is added to the playlist. The songs in the final playlist recommended by our system are about one hour long total, or equivalently 15 songs. Suppose the user has 10 favorite artists. Then, the additional five songs are obtained by randomly selecting five from the 10 second-ranked candidate artists and adding their most popular songs to the playlist. In other words, only one song is selected from each candidate artist. If the favorite artists of a user is more 15, then the 15 candidate artists are chosen randomly. In our current implementation, the popularity of a song is obtained by checking the total number of views of the song on YouTube [24] since it was uploaded. We can also define the popularity of the song by the frequency of views.

E. Post Processing

To ensure that only novel artists are recommended, a candidate artist is removed from the playlist if it is one of the favorite artists of the user. In addition, if the amount of *dislikes* for the song in the playlist is larger than the amount of *likes*, it is considered an outlier and replaced by the next popular song by the artist.

Algorithm 1 is the procedure of our system.

IV. Evaluation

The proposed music recommendation system is intented for real applications targeting young audience. Therefore, the performance of the system is evaluated by subjects recruited from university campus. The test condition and results are described in this section.

Algorithm 1: Automatic playlist generator

Input: A finite preferred-artists set
$A = [a_1, a_2, \ldots, a_n]$
Output: A 15-songs playlist $PL = [pl_1, pl_2, \ldots, pl_{15}]$

for $i \leftarrow 1$ ***to*** n **do**
  [S, Sim] = SearchLastfm($A(i)$)
  P_a = SearchGoogle($A(i)$)
  P = SearchGoogle(S)
  $P_{norm} = logP/logP_a$
  $L(i,:)$ = Likelihood($A(i), Sim, P_{norm}$)
end
$List = argartist$SelectTop15Score(L)
PL = GetMostPopularSong($List$)
return PL
def Likelihood($A(i), Sim, P_{norm}$)
for $j \leftarrow 1$ ***to*** $|Sim|$ **do**
  $LL = Sim(1 - P_{norm})$
  Sort(LL,descend)
  return LL
end
def GetMostPopularSong($List$)
for *a in List* **do**
  s = Get most popular song on Youtube
  PL = [PL, s]
end
return PL

TABLE I
EXPERIMENT RESULT OF OUR SYSTEM

Score	User-preference	Novelty
5	28.89%	72.84%
4	34.57%	10.59%
3	24.39%	8.24%
2	8.81%	4.31%
1	3.33%	4.02%

TABLE II
EXPERIMENT RESULT OF SPOTIFY

Score	User-preference	Novelty
5	23.75%	53.25%
4	35.36%	8.46%
3	28.96%	19.20%
2	9.76%	8.24%
1	2.17%	10.85%

A. *Dataset and Evaluation Method*

Spotify Radio, which is a popular music recommendation system, is used as the benchmark in this test. The system takes favorite artists as input, just like ours. Its criterion for music recommendation is based on user preference. In contrast, the criteria of our system for music recommendation is based on both artist novelty and user preference. Despite the basis of recommendation is different, we use it for performance comparison because it is the state-of-the-art system.

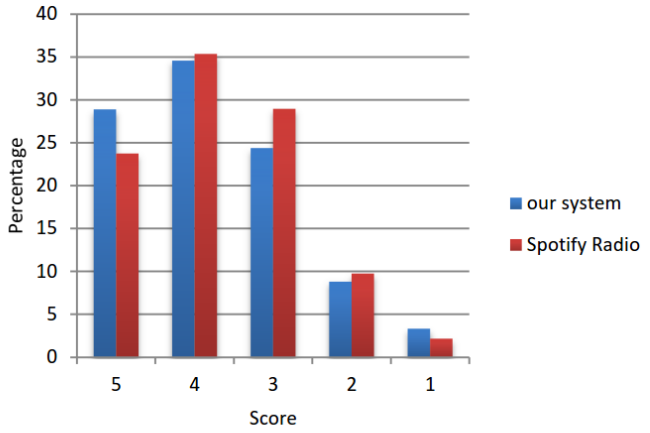

Fig. 2. Comparison of preference performance between out system and Spotify Radio.

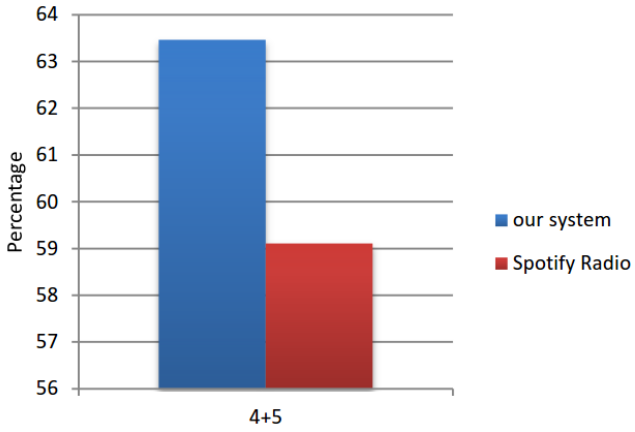

Fig. 3. Percentage of the like songs (preference score ≥ 4).

We recruit more than 106 subjects from our campus to evaluate the proposed music recommendation system. These subjects are asked to provide 7 to 10 favorite artists as input, based on which our system and Spotify Radio both generate a playlist of 15 songs for each subject. To make a fair comparison, the 30 songs are combined into a final playlist in random order. In addition, we do not inform the subjects of the fact that these songs come from two different music recommendation systems. If the two systems recommend the same song, it would appear in the playlist twice. In this case, we ask the user to give the same score to the two songs.

For the purpose of performance evaluation, we design a questionnaire to collect the subjects' opinions of each recommended song in two aspects: preference and novelty.

The subjects are asked to give one of the following five preference scores for each song:

- 5: The song is awesome. I will listen to it again.
- 4: Nice. I might listen to it again.
- 3: The song is okay, but I may not listen to it again.
- 2: Nothing special. No comment.
- 1: The song is terrible.

Songs that are rated 4 or above are considered to be in the *like* category, whereas those below 3 are considered to be in the *dislike* category. Here, we exclude score 3 for either category,

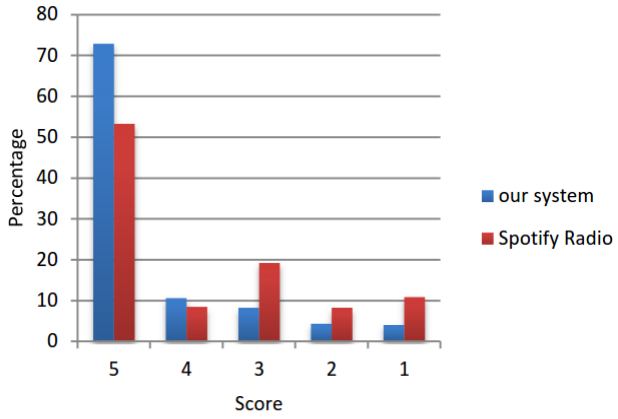

Fig. 4. Comparison of novelty between our system and Spotify Radio.

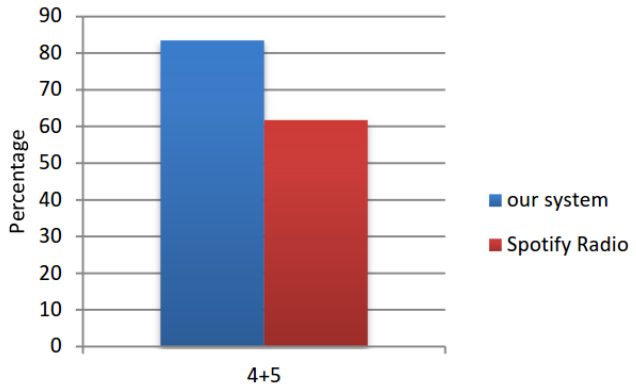

Fig. 5. Percentage of the new songs (novelty score ≥4).

because it is a neutral score and does not suggest a like or dislike preference.

The subjects are also asked to give one of the following five novelty scores for each song:

- 5: I've neither heard of the artist nor the song.
- 4: I know the song, but I haven't heard of the singer.
- 3: I know the singer, but I haven't heard of this song.
- 2: I know both the singer and the song.
- 1: I am quite familiar with the singer and this song

Songs that are rated 4 or above are considered to be in the *new* category, whereas those with score 2 or 1 are considered to be in the *known* category. Similar to the classification of the preference scores, a novelty score of 3 is excluded from the *new* and the *known* categories, because it is a neutral score.

B. Experimental Results

The experimental results of our system and Spotify Radio are summarized in Tables I and II. The scores are also presented in the form of figures (Figs. 2 and 4).

1) *Recommendation Accuracy*: A comparison of the preference results between our system and Spotify Radio is shown in Fig. 3. For our system, 63.47% of the songs are rated 4 or 5 by subjects and that 12.14% of songs are rated 1 or 2. For Spotify Radio, 59.11% of the songs are rated 4 or 5, and 11.93% of songs are rated 1 or 2. It can be seen that the performance of our system in preference is slightly better than Spotify Radio.

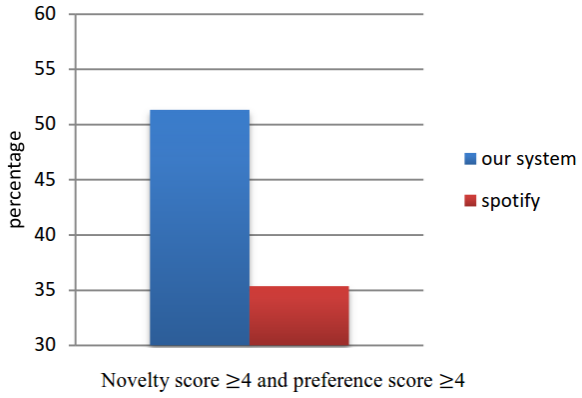

Fig. 6. Percentage of new and like songs (novelty score ≥4 and preference score ≥4).

Recommendation Novelty: Fig. 4 shows a comparison of the novelty between the two systems. The result indicates that 72.84% of the songs recommended by our system are totally unheard of by users, while Spotify Radio achieves 53.25% only. The result in Table II also shows that 19.01% of the songs recommended by Spotify Radio are known to the subjects, while only 8.33% of the songs recommended by our system are known to the subjects. Clearly, our system generates songs with much higher novelty than Spotify Radio.

2) *Discussion:* Compared to Spotify Radio, our system obtains fairly high novelty performance, while its preference performance is at the same level. However, the percentage of songs with the lowest preference score in our system is 3.33%, which is 1.17% higher than Spotify Radio. This is a result of the tradeoff between novelty and preference. Nonetheless, we believe the slight cost in preference paid for the big gain in novelty is worthwhile. This is consistent with the objective of our design to go for a music recommendation system based on artist novelty and similarity. Note that the overall performance of our system is greater than Spotify Radio, as shown in Fig. 6. Among the songs recommended by our system, 51.32% of them are new and like, whereas only 35.36% of songs recommended by Spotify Radio are new and like, as rated by the subjects.

V. CONCLUSION

In this paper, we have presented a personalized music recommendation system that recommends artists who are not yet famous. The performance of the system is evaluated by more than 100 subjects. It achieves high novelty performance and similar preference performance compared to the popular recommender Spotify Radio. The low complexity of the system makes it desirable for real-world applications and easy to integrate with existing music services. As future work, we plan to implement the collaborative filtering method and integrate it with our system to address the over-specialization issue of content-based filtering.

ACKNOWLEDGMENT

This work was supported by grants from the National Science Council of Taiwan under Contracts NSC100-2221-E-002-198-MY3 and NSC-101-2622-E-002-0060-CC2 and from the National Taiwan University under Contracts NTU-CESRP-103R7609-2.

REFERENCES

[1] (2014) Spotify Ltd. [Online]. Available: https://www.spotify.com/
[2] (2014) Last.fm. [Online]. Available: http://www.last.fm/home
[3] S. M. McNee, J. Reidl, J. A. Konstan, "Being accurate is not enough: how accuracy metrics have hurt recommender systems," in *Chi'06*, 2006, p. 1097-1101.
[4] T. Zhou, Z. Kuscsik, J. Liu, M. Medo, J. R. Wakeling, Y. Zhang, "Solving the apparent diversity-accuracy dilemma of recommender systems," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, pp. 4511-4515, Mar. 2010.
[5] N. Hurley, M. Zhang, "Novelty and diversity in top-N recommendation—analysis and evaluation," *ACM Transactions on Internet Technology*, vol. 10, pp. 14:1-14:30, Mar. 2011.
[6] Celma, Òscar, and Pedro Cano. "From hits to niches? or how popular artists can bias music recommendation and discovery. " *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*. ACM, 2008.
[7] M. J. Pazzani, D. Billsus, "Content-based recommendation systems," *The Adaptive Web*, pp. 325-341, 2007.
[8] C. Christakou, S. Vrettos, A. Stafylopatis, "A hybrid movie recommender system based on neural networks," *Artificial Intelligence Tools*, vol. 16, pp. 771, Oct. 2007.
[9] Yi Zhang, J. Koren, "Efficient Bayesian hierarchical user modeling for recommendation system," in *SIGIR'07*, 2007, p. 48-54.
[10] R. J. Mooney, L. Roy, "Content-based book recommending using learning for text categorization," in *Proceedings of DL*, 2000, pp. 195-204.
[11] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, M. Slaney "Content-based music information retrieval: current direction and future challenges," *Proceedings of IEEE*, vol. 96, pp. 668-696, Apr. 2008.
[12] M. Balabanovic, Y. Shoham, "Fab: content-based, collaborative recommendation system," *Comm. of the ACM*, p. 66-72, Mar. 1997.
[13] J. S. Breese, D. Heckerman, C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of UAI*, 1998, p. 43-52.
[14] J. B. Schafer, D. Frankowski, J. Herlocker, S. Sen, "Collaborative filtering recommender systems," *The Adaptive Web*, pp. 291-324, 2007.
[15] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of WWW*, 2001, p.285-295.
[16] G. Linden, B. Smith, J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE International Computing*, vol. 7, pp. 76-80, Jan./Feb. 2003.
[17] T. Hofmann, "Collaborative filtering via Gaussian probabilistic latent semantic analysis," in *Proceedings of SIGIR*, 2003, p. 259-266.
[18] T. Hofmann, "Latent semantic models for collaborative filtering," *ACM Transactions on Information Systems*, vol. 22, pp. 89-115, Jan. 2004.
[19] H. Kim, A. Ji, G. Jo, "Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation," *Electronic Commerce Research and Applications*, vol. 9, pp. 73-83, 2010.
[20] T. Miranda, M. Claypool, A. Gokhale, T. Mir, P. Murnikov, D. Netes, M. Sartin, "Combining content-based and collaborative filters in an online newspaper," in *Proceedings of ACM SIGIR Workshop on Recommender Systems*, 1999.
[21] R. Burke, "Hybrid recommender systems: survey and experiments," *User Modeling and User-adapted Interaction*, vol. 12, pp. 331-370, Nov. 2002.
[22] (2014) Last.fm api. [Online]. Available: http://www.last.fm/api
[23] (2014) Facebook. [Online]. Available: https://www.facebook.com/
[24] (2014) Youtube. [Online]. Available: https://www.youtube.com/