Music Recommendation in the Personal Long Tail:
Using a Social-based Analysis of a User's Long-Tailed Listening Behavior

Kibeom Lee
Graduate School of Culture Technology, KAIST
Daejeon, Korea
kiblee@kaist.ac.kr

Woon Seung Yeo
Graduate School of Culture Technology, KAIST
Daejeon, Korea
woon@kaist.ac.kr

Kyogu Lee
Department of Digital Contents Convergence, Seoul National University
Seoul, Korea
kglee@snu.ac.kr

ABSTRACT

The online music industry has been growing at a fast pace, especially during the recent years. Even music sales have moved from physical sales to digital sales, paving the way for millions of digital music becoming available for all users. However, this produces information overload, where there are so many items available due to, virtually, no storage limitations, it becomes difficult for users to find what they are looking for. There have been many approaches in recommending music to users to tackle information overload. One successful approach is collaborative filtering, which is currently widely used in commercial services. Although collaborative filtering produces very satisfying results, it becomes prone to popularity bias, recommending items that are correct recommendations but quite "obvious". In this paper, a new recommendation algorithm is proposed that is based on collaborative filtering and focuses on producing novel recommendations. The algorithm produces novel, yet relevant, recommendations to users based on analyzing the users' and the entire population's listening behaviors. An online user test shows that the system is able to produce relevant and novel recommendations and has greater potential with some minor adjustments in parameters.

Categories and Subject Descriptors

I.1.2 [**Computing Methodologies**]: Algorithms – *Nonalgebraic algorithms, analysis of algorithms*

General Terms

Algorithms

Keywords

Recommender systems, collaborative filtering, music recommendation

1. INTRODUCTION

With advances in the Internet, lower hardware costs, increasing peer-to-peer networks, and the popularity of high-storage portable media players, the online music industry has been growing rapidly, especially during the past few years. Gradually, music sales have moved from physical album sales to digital sales from online stores. Currently, these services offer millions of tracks to users, the catalog growing rapidly in size compared to the size when the services were first announced. For instance, Amazon offered over 2 million songs to users when the music service launched, but now offers over 11.8 million songs as of 2010. Some notable online music stores, including Amazon, are Amazon MP3 (11,000,000+ songs), iTunes Store (12,000,000+ songs) and Rhapsody (9,000,000+ songs). Apart from music stores, there are also music streaming services that offer millions of songs, such as Lala (8,000,000 songs), Spotify (8,000,000 songs), and Last.fm (7,000,000 songs).

These large numbers of songs available to users are a result of the Long Tail business model [1], contrary to only products that were in demand being sold in stores. However, as a result, although paradoxical, users have ended up listening to less music now that so much is available, simply because it is hard to find new and relevant music. For instance, digital track sales surpassed the 1 billion sales mark in 2008. However, the Top 200 digital tracks alone accounted for 17% of the entire track sales (184 million sales) [2].

WOMRAD 2010 Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2010 (Barcelona, SPAIN)

2. RELATED WORK

2.1 Collaborative Filtering-based Recommender Systems

One of the earliest recommender systems based on collaborative filtering is Tapestry [3]. Stemming from the need to handle increasing numbers of emails, Tapestry used explicit opinions of people in a relatively small group, such as an office workgroup, to filter out incoming email for a given user. However, a drawback of this system was that users had to be familiar with the preferences and opinions of other people in their network, which is why Tapestry worked on small networks like the office.

A more general collaborative filtering approach was developed by Resnick et al. called GroupLens [4]. The basic idea behind GroupLens, which aimed to help users find news articles amongst the vast available numbers, was that "people who agreed in the past will probably agree again". Using this heuristic, the GroupLens system was able to predict the ratings of certain news articles by a given user. An advantage that this provided was that the collaborative filtering could be scaled, unlike Tapestry, because a user was not required to actually know other users that had similar preferences to him. This was done by the system, which gathered information on the ratings of users, naturally

creating another advantage of users being anonymous inside the whole system.

Research related to, and including, the above studies focused on filtering a vast amount of text, which were in forms of emails, news, and messages, to those that were worth reading. Items would be given to the user with their prediction scores, aiding the user in which item to read next. The next wave of studies focused on a more direct approach in recommending items.

Ringo was a system developed to provide personalized music recommendations [5]. It maintained a user's profile, a history of ratings on various artists that were essentially explicit labelings on which artists the user does or does not enjoy listening to. These profiles were matched by the system to calculate recommendations on which artists had the highest probabilities of being liked by the user.

While Ringo was focused on music items, Bellcore's recommender system focused on movies [6]. Like Ringo, it used a database of movie ratings by users and matched rating profiles to provide recommendations by finding similar users and the movies that they had watched and rated positively. Tests on the reliability of the recommender system showed that three out of every four recommendations would be rated highly by the user, and also showed that the system produced extremely more accurate recommendations compared to nationally-known movie critics.

While there were numerous advances and algorithms related to collaborative filtering since then, the most well-known collaborative filtering system today, however, is probably the system used in Amazon.com, an electronic commerce company that sells books, movies, music, etc. Amazon.com offers recommendations on items that are similar to the item being purchased, rather than finding similar users and then recommending the items those users have purchased. This method, which is called item-to-item collaborative filtering, scales to extremely large datasets and generates satisfiable results.

2.2 Collaborative Filtering-based Recommender Systems for Music

Although the collaborative filtering-based approaches above were designed on specific items, the algorithms can be generalized and applied to music recommendation. Hence, the results of such algorithms applied to music are not much different than applied to the original items.

Apart from recommender systems that use data on the ratings and/or purchases of items, there are other collaborative filtering-based recommender systems that take advantage of metadata produced by users that are found in music.

[7] presents some examples of metadata used in such algorithms, which include reviews, lyrics, blogs, social tags, bios, and playlists. Examples of commercial services that use such approaches are Rate Your Music (reviews), The Hype Machine (blogs), last.fm (social tags), and playlist.com (playlists).

Social tags, a representative product of online collaboration, has been used heavily in music recommendation systems. Hu and Downie explored the mood metadata associated with songs and their relationships with music genre, artist, and usage metadata [8]. They found that the genre-mood relationships and artist-mood relationships showed consistencies, showing the potential of being utilized in automated mood classification tasks. Eck et. al proposed a method for generating social tags for music that lack such tags [9]. Audio features of songs were analyzed and mapped to tags, using a set of boosted classifiers. These were then utilized on untagged songs, populating them with the associated social tags depending on the musical content. This enables unpopular songs and/or new songs that have no social tags to be used in music recommenders that use a social algorithm. It also tackles the cold start problem, a problem found in collaborative filtering-based recommender systems. Symeonidis et. al analyzed social tags in order to tackle the problem of the multimodal use of music [10]. They developed a framework that modeled users, tags, and items, altogether. This was then used in recommending musical items (artists, songs, and albums) to users by performing latent semantic analysis and dimensionality reduction according to each user's multimodal perception of music. Levy and Sandler inspect the seemingly ad hoc and informal language of tagging as a high-volume source of semantic metadata for music. Results show that tags establish a low-dimensional semantic space, being extremely polished at the track level, especially by artist and genre. Using these results, the authors also introduce an interface for users to browse by mood, through a two-dimensional subspace that represents musical emotion.

Celma introduces a system that recommends music and the relevant information associated with the recommended music [11]. The proposed system uses the *Friend of a Friend* and RSS vocabularies for creating recommendations, taking in consideration the user's musical tastes and listening habits. The FOAF project provides protocols and a language to describe homepage-like content and social networks, ultimately providing the proposed system with the user's profile. The RSS vocabulary provides the system with syndicated content, which includes data such as new album releases, album reviews, podcast sessions, upcoming gigs, etc. Thus, the proposed system improves the existing recommendation systems by understanding the users through psychological factors (personality, demographic preferences, socioeconomics, situation, social relationships) and explicit music preferences.

3. LIMITATIONS OF COLLABORATIVE FILTERING

3.1 Popularity Bias

Collaborative filtering-based recommender systems produce good results and are used widely in commercial services such as Amazon.com and Last.fm. However, collaborative filtering has some common limitations that occur naturally due to its roots lying in the wisdom of crowds. One of the largest problems of collaborative filtering is popularity bias [12, 13].

This happens when a popular item is associated with many other related items. Users that interact with these items are then recommended the popular item. The system recommends the popular item often, leading to item purchases (or any other form of positive input from user) and as this item is purchased more, it is also recommended more. This loop, in which the rich become richer, creates popularity bias.

Naturally, as a result of the above feedback loop, the recommender system tends to bias its recommendations towards popular items. Thus, the recommendations lose their novelty [12, 13] and make it extremely difficult to recommend lesser-known artists.

In Amazon.com, in which collaborative filtering is heavily used, the popularity bias can be seen when viewing the recommendations that are offered when searching for popular items. For instance, the 98 recommendations that appear when searching for Harry Potter includes The Da Vinci Code, To Kill a Mockingbird and 28 other Harry Potter books and DVDs. In the case of music, searching for The Beatles' Revolver album results in 33 albums from The Beatles, out of a total of 97 recommendations, as shown in Figure 1. The other recommended items show well-known artists that user's, who are interested in The Beatles, will most likely have heard of already such as The Rolling Stones, Led Zeppelin, and Neil Young. These recommended artists are *correct* recommendations but fail to be novel recommendations.

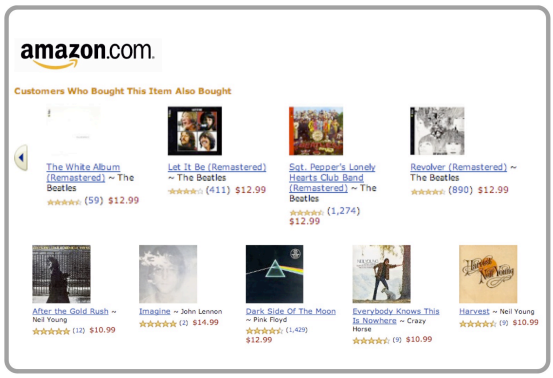


Figure 1. Recommendations from Amazon.com, which are all quite "obvious" recommendations, although they are correct recommendations.

Due to this popularity bias, a large portion of the recommended items result in obvious recommendations that may be relevant to easy-going, casual listeners, but not so helpful for enthusiastic music listeners, who have a high probability of already being knowledgeable on the artists being recommended.

The number of high quality, or "correct", recommended items that are produced with collaborative filtering is verified by [14]. However, the problem of popularity bias was also verified as the amount of novel recommendations given to a user was the lowest for collaborative filtering in an experiment comparing collaborative filtering, content-based, and hybrid methods [14]. Thus, it was confirmed that collaborative filtering results in less percentage of novel songs but of higher quality.

4. ALGORITHM

In this section, we provide an algorithm that is based on collaborative filtering, yet overcomes popularity bias, a natural problem that arises from CF. Also, the algorithm focuses on providing recommendations that are novel to the user, while also remaining relevant.

To implement this algorithm, user data from Last.fm, an Internet service that provides users with streaming music via radio stations, was used. Reasons for selecting Last.fm was the readily available developer API and the various, massive amount of data that was available such as user playlists, playcounts for artists and individual songs, artist information, song information, and most importantly, the worldwide popularity of the site.

4.1 Concept of Recommendation Algorithm

4.1.1 Changing Perspectives on Novel Recommendations

While the goal of recommenders in general is to provide recommendations that are novel and relevant to the user, as stated beforehand social-based recommendations, although relevant, fail in providing novel recommendations to users. In contrast, content-based recommender systems work better in providing novel recommendations because they are not affected by popularity or any other social influence [15].

Another method to provide novel recommendations to users is to use the long tail popularity distribution of the artists [7]. This idea can be applied to both content-based and social-based algorithms. Content-based algorithms can use the long tail distribution to recommend similar items based on content-analysis and also found in the tail portion of the distribution. For social-based algorithms, or collaborative filtering, the idea can be applied by first obtaining the full list of recommendations and then removing the recommendations that lie in the head portion of the distribution. This would result in recommendations being novel to the user, since it is unlikely that artists residing in the tail portion of the distribution are known to the user.

However, although strictly recommending artists from the long tail and avoiding recommending those that are obvious (those that are located in the head portion of the distribution) have a high probability of being novel recommendations, we need to take in consideration that novel recommendations are relative to the user. In other words, it is naive to assume that the user will be aware of certain artists just because they are in the head portion of the long tail distribution. Thus, the fact that even popular artists have a possibility of being novel recommendations to certain users must not be overlooked.

4.1.2 User Listening Behavior

As shown in Figure 2, which shows a random Last.fm user's playlist in descending order of playcount, the listening behavior shows a distribution that is similar to that of long-tail distributions. Users tend to listen to an extremely small portion of their playlists often while the remaining songs seldom get played. Due to the data available, which is the top 500 played songs in the user's playlist, all of the songs in the graph are played at least once.

Figure 2. The listening behavior of a user and his/her entire playlist. Although not exact, the graph shows a long-tailed distribution where the majority of tracks are seldom played.

4.1.3 Defining Experts and Novices

Using this long-tailed distribution of users' listening behaviors, the users can be divided into two groups: experts and novices. Here, users are considered "experts" regarding the songs/artists that they listen to often, i.e. songs/artists that lie in the head portion of the long-tailed listening behavior. On the other hand, users are considered "novices" regarding the songs that reside in the tail portion.

4.1.4 The Mystery of Unpopular "Loved" Songs

Last.fm provides users with an option to mark songs "loved" (Figure 3). This kind of feedback from users explicitly shows that a user enjoys a particular song. One would expect that these "loved" songs would all lie in the head portion of the listening behavior distribution. However, these songs that are marked "loved" can be found scattered throughout the entire distribution. Here, a paradox can be found: Why are some songs marked "loved" lying at the tail end of the playcount distribution? One would assume that a "loved" song would have a high playcount, but a quick inspection shows that this is not the case. Thus, an assumption that is made here, a key assumption in this algorithm, is that songs are marked "loved", yet remain in the tail, because the user is unfamiliar with that song/artist/genre, i.e. is a novice, but happened to stumble upon that particular song and liked it.

Figure 3. The "tail" portion of a random user's playlist. There are two songs marked "loved" by the user, but have only been played three times.

Among the 21,688 users whose data was used for the algorithm, 78.3%, or 16,973 users, used the "love" function provided in Last.fm. Among the 16,973 users who utilized the "love" function, 77.8% of the users had "loved" songs in the tail portion of their playlist's song distribution sorted by playcount.

Upon closer inspection of the random user in Figure 3, the songs/artists in the "head" portion came from various genres such as electronic, hip-hop, and reggae. What they did have in common, however, was that they were all German artists, including the user herself. Looking at the songs that were marked "loved" but were not played often, we can see that they too come from different genres, but are both artists from the U.S.

The previously mentioned assumption that fuels this algorithm was made after observing such occurrences in users' playlists. According to our assumption, we assume that the user, who is German, is a novice in artists from the U.S. and stumbled across several songs that she liked. However, she did not get to venture similar songs and/or artists because she was unaware of which artists/songs were similar.

4.1.5 The Big Picture

Once the basic assumptions are made and the new definition of novices and experts are established, the concept of the recommendation algorithm can be explained. As shown in Figure 4, recommendations can be made to novices of certain song sets using the information that can be obtained by a group of experts that have those song sets in the head portion of their listening behavior distribution.

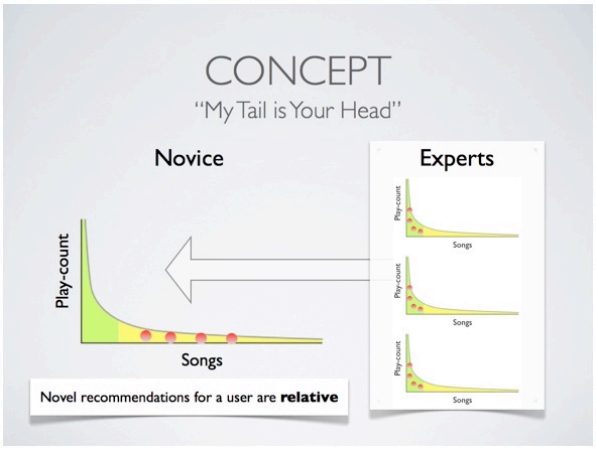


Figure 4. The overview of the algorithm showing the concept of novices and experts.

By using the listening behavior of experts to provide recommendations to novices, the recommended items will be novel to the user, contrasting to other recommendation systems that simply recommended artists/songs from the tail of the popularity distribution of items. In other words, while remaining novel to the specific user, the recommended items may or may not be in the far, unpopular end of the popularity distribution. In fact, even popular items that reside in the head of the popularity distribution may be recommended, but the user may not be aware of the recommended item since the recommendations were based on the user's tail portion of her listening behavior distribution, in which the user was considered a novice.

In addition to being novel recommendations, the recommended items will also be relevant to the user since the recommendations were found using songs that the user had marked "loved", explicitly stating the user's view on that particular item, and then using collaborative filtering to find experts on those "loved" songs to find relevant recommendations.

4.2 Data

User data was collected in order to test the algorithm and evaluate the results of the recommendations from early March to late April in 2010. Data was collected from the Last.fm website using a custom web crawler and the Last.fm API. The user data that was collected included the songs that the user had listened to overall, meaning the songs that the user listened to from the day he/she registered at Last.fm up until the day the data was collected. It also included the playcount for each song, song title, artist name, user ID, rank, and whether it was marked "loved" or not. The data that was collected is summarized below in Table 1.

Table 1. Summary of amount of data collected

Data	Count
Users	21,681
Unique Songs	2,001,324
Songs from All Playlists	9,073,681

4.2.1 Last.fm API

All the collected information, except the playlist history, was gathered via the Last.fm API. Although the algorithm could have queried the information in real-time, it was decided that having local data would facilitate in quicker results. After fetching the data, we had song titles and corresponding artist names of approximately 2 million songs.

In addition to the user and song data collected with the Last.fm API, artist popularity was also measured indirectly via the API. Because the Last.fm API did not provide the artist ranking directly through the API, we had to collect the number of Listeners and Plays, which were offered through the API. By having the Listeners and Plays of a given artist, we would be able to determine the overall ranking of popularity of the artists. This will be further explained in the next section.

4.2.2 User Data Crawler

Unfortunately, the Last.fm API query for a given user's listening history returns the top 50 songs ordered by playcount. This was not adequate enough since the algorithm needed the entire playlist in order to utilize the long tail of the playcount distribution.

In order to solve this problem, a custom crawler was implemented to collect the users' listening history (referred to as 'playlist' in this paper) and playcount information. Although this returned a maximum of 500 results (Last.fm displays only top 500 songs in the playlist), the data was adequate to be divided into the short head and long tail and used in the algorithm.

Data on a total of 21,681 random users were crawled. The playlists and the according information were also stored for each user, resulting in 21,681 playlists with a total of 9,073,681 songs. Because playlists from different users contain lots of duplicate entries, the number of unique songs that were crawled, as stated above, was 2,001,324 unique songs.

4.3 Algorithm

As shown in Listing 1, the user that will receive the recommendations, whom we will call "novice" according to the algorithm's concept, is given as input to the algorithm. Then, the listening behavior for the novice is retrieved using data available at Last.fm. As long as the user is not a new user and has been listening to his/her playlist, the playcount distribution of his/her playlist is more than likely to show a long-tailed distribution, in which a small set of songs have been listened with a heavily biased frequency while the remaining songs listened only occasionally. Since we are interested in the songs/artists that the given user is a novice on (i.e. songs marked "loved" in the long tail), we discard the head portion of the distribution and from the remaining songs, which are songs in the tail portion, we discard all songs except those that are explicitly labeled "loved" by the novice. These remaining songs, denoted by 'S_1', will be the song set that will be used to create recommendations.

Next, using the listening behavior of the other users from our database, we find those that listen to the songs in song set S. In other words, we find the "experts" on song set S by finding users that have a subset of song set S in the head portion of their listening behavior distribution. If such users exist, we compare the songs in the "head" of their playcount distribution with song set S and use the remaining, non-overlapping songs as recommendation candidates and assign the weight for those items according to the strength of the match between the songs in the expert's "head" and song set S.

```
begin Recommendations REC (aGivenUser U1);
  do
    Result R1 := retrieveListeningBehaviorDistribution(U1);
    SongSet S1 := getSongsInLongTail(R1);
    S1_loved := filterLovedSongs(S1);
    for i := 2 to n (n: number of users) step 1 do
      Result Ri := retrieveListeningBehaviorDistribution(Ui)
      SongSet Si := getSongsInHead(Ri);
      if (Si ∩ S1 ≠ ∅) do
        CandidateSongSet CSi := (Si ∪ S1) – (Si ∩ S1);
        incrementWeight(CSi);
        REC += CSi;   od
      od
    od
    printRecommendations();
```

Listing 1. Pseudoalgorithm for proposed recommender system.

These recommendation candidates are accumulated in the global song set REC, and the weight of the candidate are incremented as they are recommended to REC. Finally, the recommendations are given to the user in the order of their weights.

4.4 Parameters

The algorithm is quite flexible as it has many parameters that can be changed, which greatly influences the recommended items to the user. Parameters that play a crucial role in the overall quality of the recommendations include:

- The size of the "head" of experts
- The size of the "tail" of novices
- Weights of recommended items

4.4.1 Expert Parameter

The parameter that influences the outcome most is the size of the "head" portion of the expert's listening behavior distribution. For example, if the value for this parameter is set to "10", a user is considered an expert only if the top ten songs that s/he listened to contains any number of songs from the set of songs that are marked "loved" in the novice's "tail" portion of his/her listening distribution. In other words, this parameter determines the qualification strictness on which users are considered experts.

The lower the value, the harder it is for a given user to be considered an expert. Also, as the value is lower, the resulting recommendations are more novel, in contrast to when the values are higher, in which the resulting recommendations become those that are well-known. As the value is set higher, the recommendations represent those that are from the existing music recommendations that are offered using traditional collaborative-filtering methods.

4.4.2 Novice Parameter

The parameter that can be varied for the novice users is the size of the "tail" portion of the novice's listening behavior distribution.

Opposite of the expert parameter, the novice parameter sets the range of songs in the user's playlist that the user is a novice on. Using loved songs that lay near the "head" portion may result in songs that the user is aware of, leading to the recommendations being less novel to the novice. However, this parameter does not have as much influence as the expert parameter has because once the novice parameter is set, the entire range of songs are not used, but only those that are explicitly marked "loved" by the user.

4.4.3 Weights of Recommended Items

A formal set of rules and equations to assign weights to the recommended items can greatly change the songs that will be presented to the user as recommendations. This is important because it is inappropriate to present the entire collection of songs that result from the algorithm, as the number may vary depending on the two parameters above. Among the final song set that contains hundreds of candidate songs for recommendations, only a subset, namely the top N songs are presented to the user. Thus, assigning the appropriate weights for these candidates can ultimately influence the outcome of the recommended items. Currently, the algorithm uses a simple approach in which the weight is equal to the number of times a song is a member of both the head of an expert and tail of the novice.

5. USER TEST & EVALUATION

There are many ways to evaluate a recommender system, both offline and online. A common online method to evaluate a recommender system is to generate test sets to be evaluated later [16]. Another popular method is to use cross-validation, in which the data is partitioned and used as test sets [17].

5.1 Difficulties in Evaluating Novel Recommendations

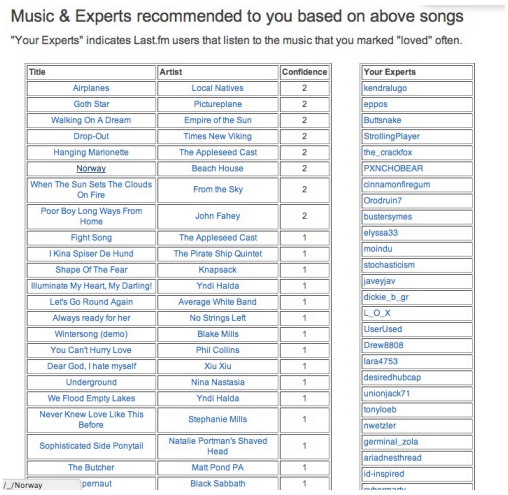
However, offline evaluations are not appropriate for recommender systems where the recommendations of novel items are important. This is because when a truly novel item is actually recommended to a user, meaning that the user does not already know about this item, it is extremely difficult for the user to evaluate the unknown item without providing any additional information [18]. Because of this, measuring novelty in the recommended items is a rather challenging task, leaving no option but to carry out live user studies where the users explicitly indicate whether the provided recommendations were novel or not [19].

Thus, in order to measure the novelty and relevance of the recommended items, an online user test was carried out using a fully functional website, including a section for explicit user feedback regarding the recommendations given to the users.

5.2 Design

A fully functional website was created in order to perform an online evaluation of the recommendations for random users. On the website, a user has to sign-up and input his/her Last.fm ID. After receiving a new ID, the server runs the recommendation algorithm on that particular Last.fm ID. Meanwhile, the user was requested to come back shortly afterwards, while the recommendations were being processed. The algorithm had to be run in real-time online because of the nature of it being heavily dependent on the user information. Also, pre-calculating the recommendations for users in the local database offline and then providing them online was unrealistic as the probability that a new user would also be one that was pre-calculated was extremely low. When the user returns, he/she is presented with two sets of recommendations.

Recommendation Set 1 was the results of the algorithm with the Expert Parameter, the parameter that determines the size of the "head" portion of the expert, set to 5. A value of 5 for the Expert Parameter means that the algorithm is being very strict about which users are qualified to be experts. This produces dense novel items. Recommendation Set 2 was the results with the Expert Parameter set to 10. A value of 10 tends to mix novel recommendations and well-known recommendations, so is more of a general setting that aims to resemble recommendations from Last.fm. After the user views the recommendations, a survey page was available to provide explicit feedback on the quality of the recommendations given to them.


Music & Experts recommended to you based on above songs

"Your Experts" indicates Last.fm users that listen to the music that you marked "loved" often.

Title	Artist	Confidence
Airplanes	Local Natives	2
Goth Star	Pictureplane	2
Walking On A Dream	Empire of the Sun	2
Drop-Out	Times New Viking	2
Hanging Marionette	The Appleseed Cast	2
Norway	Beach House	2
When The Sun Sets The Clouds On Fire	From the Sky	2
Poor Boy Long Ways From Home	John Fahey	2
Fight Song	The Appleseed Cast	1
I Kina Spiser De Hund	The Pirate Ship Quintet	1
Shape Of The Fear	Knapsack	1
Illuminate My Heart, My Darling!	Yndi Halda	1
Let's Go Round Again	Average White Band	1
Always ready for her	No Strings Left	1
Wintersong (demo)	Blake Mills	1
You Can't Hurry Love	Phil Collins	1
Dear God, I hate myself	Xiu Xiu	1
Underground	Nina Nastasia	1
We Flood Empty Lakes	Yndi Halda	1
Never Knew Love Like This Before	Stephanie Mills	1
Sophisticated Side Ponytail	Natalie Portman's Shaved Head	1
The Butcher	Matt Pond PA	1
/_/Norway	pernaut	Black Sabbath	1

Your Experts
kendralugo
eppos
Buttsnake
StrollingPlayer
the_crackfox
PXNCHOBEAR
cinnamonfiregum
Orodruin7
bustersymes
elyssa33
moindu
stochasticism
javeyjav
dickie_b_gr
L_O_X
UserUsed
Drew8808
lara4753
desiredhubcap
unionjack71
tonyloeb
nwetzler
germinal_zola
ariadnesthread
id-inspired

Figure 5. Screenshot of the recommended items at the user-test website. Each facet of the recommended items are linked to pages at Last.fm for supplementary information

Since the goal of the algorithm is to provide novel recommendations, there had to be an easy way for the user to evaluate the recommended items, since it is assumed that if the recommended items are indeed novel, then the user has no knowledge about the item. Thus, each recommended item was hyperlinked to the according page in Last.fm, as shown in Figure 5. Through these links, users were able to evaluate the recommended items that were novel to them by visiting the linked pages. Last.fm provides related information regarding specific songs, which include music videos, song previews, and even a radio for the song's artist. By utilizing these pages, users were able to listen to the songs that were recommended to them.

5.3 Survey

On the survey page, a set of five questions were given to the user, each regarding one of the two sets of recommendation results that were produced by the algorithm. The questions were answered on a five-point Likert item. The final question was a subjective question, asking for any comments or feedbacks on the recommendations. The questions used in the survey are shown in Table 2.

Table 2. Questions used in the user survey.

Q. 1	How would you rate the relevance of items?
Q. 2	How would you rate the novelty of the recommended items?
Q. 3	How would you rate the serendipity of the recommended items?
Q. 4	How would you rate the recommendations overall?
Q. 5	Provide any comments/feedback about the recommendations that were given to you.

6. RESULTS & DISCUSSION

A user survey was carried out online accompanying the online music recommendation service because of the difficulties in measuring novelty. A total of 24 users tested the recommendations offered to them on the website. These users were random Last.fm users that had received private messages (advertising the user test) through the Last.fm messaging system. The new recommendation system was also advertised on various Last.fm groups whose interests were in finding new music or those who were unsatisfied with current recommender systems and their quite obvious recommendations. However, because the users had to answer two surveys for two different sets, some appeared to have quit abruptly after finishing the first set. As a result, only 11 users out of 24 completed the second survey.

The private messages were sent to random Last.fm users who satisfied two conditions: 1) the user used the "loved" function with his/her playlist, 2) The last time the user logged in was not more than two weeks ago from the day the private messages were sent. Despite the advertisements and private messages, the response rate was extremely low (< 10 %). The results are shown in Figures 6-8.

Figure 6. Comparison of the relevance ratings for the two sets

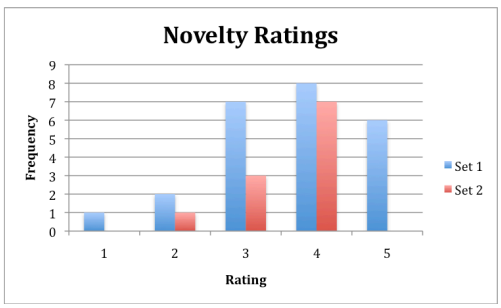


Figure 7. Comparison of the novelty ratings for the two sets

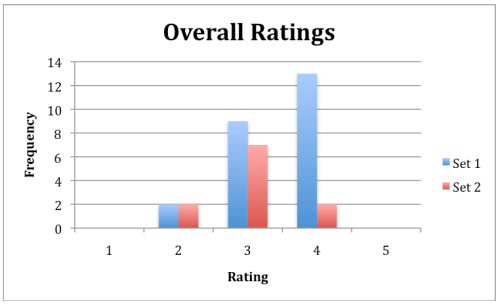


Figure 8. Comparison of the overall ratings for the two sets.

The results of the user test on the recommendations produced by the proposed algorithm are generally positive. The mean value for the relevance of the items was 3.417 (on a 5 point scale) with a confidence interval of 0.390 (with alpha value of 0.05). The mean values of novelty and serendipity were also on the positive side with 3.667 and 3.625, respectively. The confidence intervals were 0.436 (alpha = 0.05) for novelty and 0.350 (alpha = 0.05) for serendipity. The overall rating of the recommender system had a mean value of 3.458 with a confidence interval of 0.263 (alpha = 0.05). In general, the results show that the proposed system has positive ratings and could be refined to produce better results.

The proposed system was rated higher in both novelty and serendipity, compared to the second set of recommendations, which was a set of recommendations that was intended to imitate existing systems such as Last.fm.

For this study, the parameters of the system were set with values that we thought produced the desired results after several iterations of the algorithm. However, a full study focused on finding the optimal values for the parameters would be an excellent follow-up study and would greatly enhance the recommendations of the system.

The score for the novelty of recommended items could have been higher, because the algorithm did not check whether the recommended songs existed in the user's library before being offered. Thus, the user would see some artists that they were aware of. As implied above, it is quite easy to increase the percentage of novel items in the entire recommendation list: simply check whether the artist exists in the user's library and if it does, exclude it from the recommendations. However, this step was excluded from the algorithm deliberately to increase the confidence of the users on the proposed system. The basis for this was [20], in which the authors found that users liked to see familiar items in the recommendations, which ultimately led to an increase of user confidence in the system. Checking to see if the user is familiar with the recommended item would produce more "dense" novel recommendations.

Regarding the novelty of items, an unforeseen problem was revealed after the user test. One user commented, "I have most of the bands recommended on my computer, I just haven't given them much of a listen to. Grizzly Bear in particular..." The problem here is whether, in this user's case, Grizzly Bear is a novel recommendation. The user states that s/he did not listen to many of the recommended artists, although those artists were in his/her library. Because the algorithm depends on the playcount of the songs in a user's library it is totally blind to tracks that reside in the library but have a playcount of 0. Thus, it recommends songs that it believes to be novel to the user, when it could in fact

exist in the library already. Unsurprisingly, the novelty and serendipity ratings from this user were low (a score of 2 for each), but the rating on the overall system was positive (a score of 4). Clarifying such issues on what a novel item is would help improve the algorithm and the user's perception of the system.

7. FUTURE RESEARCH

The most urgent and important future work on this particular study would be to find the ideal parameter settings to produce the desired recommendations. Due to the available time frame for this study, much of the algorithm analysis including the settings of the parameters, were done manually, simply by iterating through different settings and observing the results. By finding the optimized values on parameters such as Expert Head Size, User Tail Size, and Item Weights, the quality of the recommendations in novelty and relevance would be greatly enhanced.

Work on expanding the flexibility of the algorithm can also be done, creating additional parameters that bring changes to the recommendations. More parameters would mean that the algorithm could be suited for each user's needs, bringing the possibility of creating an evermore-personalized set of recommendations.

The overall system itself could be further developed to integrate content-based analysis for better results. Although the proposed method is at its infancy, we believe that the only way to improve it further (after it has fully developed independently) will be to incorporate a content-based algorithm to improve on its remaining weaknesses as an algorithm that is based on user profiles.

8. CONCLUSION

In this paper, a novel approach to recommending unfamiliar artists relative to each user was proposed in order to tackle the problem of the high density of obvious items in the list of recommendations found in today's recommender systems. The key concept in this approach was that novel items did not always have to be items that reside in the long tail of the popularity distribution. Although novel or unfamiliar items, more often than not, do indeed reside in the long tail of the popularity distribution, it is important to acknowledge that even well-known artists could be unknown to users who are (a) interested in different genres and (b) are in different cultures and/or countries.

A system that produced recommendations was implemented and was available online for users to use and rate. The recommendations were produced using data collected from Last.fm. Results of the user surveys show that the proposed system succeeds in providing novel recommendations to users, while keeping those items also relevant. This study shows the potential of such an approach to recommending novel items, while maintaining a collaborative filtering algorithm without the support from content-based algorithms.

9. ACKNOWLEDGEMENTS

The authors would like to thank Professor Sangki 'Steve' Han at the Graduate School of Culture Technology, KAIST and Sheayun Lee for their valuable comments and feedback.

10. REFERENCES

[1] Chris Anderson. The Long Tail: Why the Future of Business Is Selling Less of More. Hyperion, 2006. ISBN 1401302378.

[2] Nielsen Soundscan, State of the Industry. *National Association of Recording Merchandisers*, 2008

[3] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. Using Collaborative Filtering to Weave an Information Tapestry. *Commun. ACM,* 35(12):61-70, 1992. ISSN 0001-0782.

[4] Resnick P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. Grouplens: An Open Architecture for Collaborative Filtering of Netnews. In *CSCW 1994,* pages 175-186.

[5] Shardanand, U. and Maes. P. Social Information Filtering: Algorithms for Automating "word of mouth". In *CHI `95*, pages 210-217.

[6] Hill, W., Stead, L., Rosenstein, M., and Furnas, G. Recommending and Evaluating Choices in Virtual Community of Use. In *CHI `95,* pages 194-201.

[7] Celma, O. and Lamere, P. If you like the Beatles you might like…: a tutorial on music recommendation. *ACM Multimedia*, pages 1157-1158, ACM, 2008.

[8] Hu, X and Downie, J. S. Exploring mood metadata: Relationship with genre, artist, and usage metadata. *,* September 2007.

[9] Eck, D., Lamere, P., Bertin-Mahieux, T., and Green, S. Automatic generation of social tags for music recommendation. In *Advances in Neural Information Processing Systems 20.* MIT Press, 2008.

[10] Symeonidis, P., Ruxanda, M. M., Nanopoulos, A., and Manolopoulos, Y. Ternary semantic analysis of social tags for personalized music recommendation. *ISMIR*, pages 219.

[11] Celma, O. Foafing the music: Bridging the semantic gap in music recommendation. In *Proceedings of the 5th International Semantic Web Conference,* pages 927-934, Springer, 2006.

[12] Celma, O. and Herrera, P. A new approach to evaluating novel recommendations. In *RecSys `08:* pages 179-186, New York, 2008.

[13] Celma, O. and Cano, P. From hits to niches?: or how popular artists can bias music recommendation and discovery. In *NETFLIX '08*: *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, pages 1-8, New York, NY, USA, 2008.

[14] Celma, O. *Music Recommendation and Discovery in the Long Tail.* PhD thesis.

[15] Pampalk, E. and Goto, M. Musicrainbow: A new user interface to discover artists using audio-based similarity and web-based labeling. *ISMIR*, pages 367-370, 2006.

[16] Duda, R. O. and Hart, P. E. *Pattern classification and scene analysis.* New York, 1973.

[17] Stone, M. Cross-validatory choice and assessment of statistical predictions. *Roy. Stat. Soc.,* 36:111-147, 1974.

[18] Herlocker, J. L., Konstan, J. A., and Riedl, J. T. Evaluating collaborative filtering recommendations. In *Computer Supported Cooperative Work*, pages 241-250, 2000.

[19] Schafer, J. B., Frankowski, D., Herlocker, J., and Sen, S. Collaborative filtering recommender systems, 2007.

[20] Singha, S., Rashmi, K. S., and Sinha, R. Beyond algorithms: An HCI perspective on recommender systems, 2001