A Historical Survey of Music Recommendation Systems: Towards Evaluation

Ying Qin

Music Technology Area, Department of Music Research
Schulich School of Music
McGill University
Montreal, Canada

April 2013

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Arts in Music Technology.

Abstract

The development of the Internet and the emergence of audio compression technologies have contributed to the realization of making millions of music titles accessible to millions of users. Due to the extensive distribution of music, consumers are being presented with a problem of information overload, while the music industry is being faced with the challenge of personalized promotion and distribution. Music recommendation systems aim to ease the task of finding the music items that might interest the users by generating meaningful recommendations. The recommendation for music is different from those for books and movies, due to its low cost per item, short consumption time, high per-item reuse, highly contextual usage, and numerous item types. Understanding the patterns of music listening and consumption is important to create accurate and satisfying music recommendations.

This thesis reviews state-of-the-art music recommendation and discovery methods with the goal of presenting the historical developments in this area. Traditional music recommendation systems can be classified as one of two major kinds: collaborative filtering and content-based filtering. Recently, the research community has broadened its attention to include other aspects, such as hybrid approaches, context awareness, social tagging, music networks, visualization, playlist generation, and group recommendation.

For the evaluation of music recommendation systems, researchers or developers need to take into account properties such as accuracy, coverage, confidence, novelty, diversity, and privacy. These properties can be measured in an offline simulation, a user study, or an online evaluation. Suggestions for future work in both the design and the evaluation of music recommendation systems are given.

Sommaire

Le développement d'Internet et l'émergence des technologies de compression audio ont contribué à rendre des millions de titres de musique accessibles à des millions d'utilisateurs. Due à la large distribution de musique, les consommateurs ont des problèmes de téléchargement d'information, pendant que l'industrie de la musique faisait face aux défis de la promotion et de la distribution personnalisée. Les systèmes de recommandation de musique ont pour objectif de faciliter la tâche dans la recherche d'articles de musique qui peuvent intéresser l'utilisateur en générant des suggestions significatives. La recommandation de musique est différente de celle des livres et des films; cette différence est explicable par le bas coût de chaque article, le court temps de consommation, le haut taux de réutilisation de chaque article, l'utilisation hautement contextuelle immédiat et le grand nombre d'articles possibles. La compréhension des schémas d'écoute et de consommation de la musique est importante afin de concevoir des recommandations musicales pertinentes et satisfaisantes.

Ce mémoire passe en revue l'état des recherches sur la recommandation musicale et les méthodes de découverte, dans le but de dressez un historique des développements de ce domaine. Les systèmes de recommandation de musique traditionnels se divisent en deux catégories principales: le filtrage collaboratif ou filtrage fonde sur le contenu. Récemment, la communauté de chercheurs a élargit ces champs d'investigation en incluant d'autres aspects : tels que, les approches hybrides, la sensibilité au contexte, le marquage social, les réseaux de musique, la visualisation, la génération de liste de lecture et les recommandations de groupe.

Pour l'évaluation des systèmes de recommandation de musique, les chercheurs et les promoteurs ont besoin de prendre en compte des propriétés, comme la précision, la couverture, la confiance, la nouveauté, la diversité et la protection de la vie privée. Méthodologiquement, ces propriétés peuvent être mesurées dans des simulations hors connexion, dont l'étude du comportement des utilisateurs, ou lors d'évaluations en ligne. Des suggestions pour de futures recherches en design et évaluation de systèmes de recommandation de musique seront abordées.

Acknowledgments

I would like to sincerely thank my supervisor Prof. Ichiro Fujinaga for giving guidance, advice, and feedback for all aspects of my thesis. Thanks to Prof. Gary Scavone and Prof. Stephen McAdams for helping me formulate the proposal of this work. Thanks to the external examiner, Prof. Philippe Depalle.

Thank you to all of the members of the McGill Music Technology Area for helping me enjoy my graduate studies. Special thanks to Gabriel Vigliensoni for his help on my research. Thank you to everyone who assisted in the editing of this document. Thank you to all my friends and people who help me in one way or another during my two year's stay in Montreal.

Finally, the greatest thanks go to my parents in Liaoning province, China, for their support and encouragement in my most difficult times.

Contents

Chapter 1

Introduction

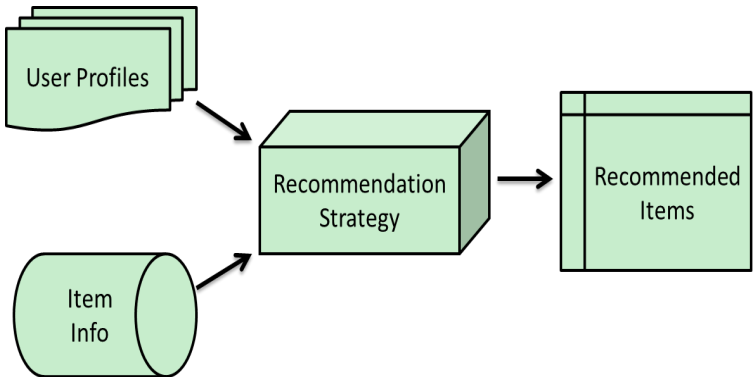
The emergence of digitization and efficient audio compression technologies, such as MP3, has realized the dream of making millions of music titles accessible to millions of users (Pachet 2003). Over the last decade, the fast development of the Internet and the Web further accelerated the distribution of music to enormous number of music listeners (Fields 2011). Due to the extensive distribution of music by major portals (e.g., *Amazon*[1], *iTunes*[2] and *Yahoo! Music*[3]) and streaming services (e.g., *Pandora*[4] and *Last.fm*[5]), consumers are facing a problem of information overload for music data (Hijikata et al. 2006). They are being presented with an increasing number of music items while the music industry is being faced with the challenge of personalized promotion and distribution. Clearly, services that help users look for their favorite music are urgently needed. Music recommendation systems are used to facilitate the task of finding appropriate items within a huge collection.

The goal of a recommendation system is to generate meaningful recommendations to users for items that might interest them (Resnick and Varian 1997; Melville and Sindhwani 2010), based on users' predefined preferences or access histories (Chen and Chen 2005). The process of a recommendation system is illustrated in Fig. 1.1. Celma (2010) pointed out that music recommendation is somewhat different from other domains, such as recommendation for movies or books. Music is unique due to its low cost per item, low consumption time, high per-item reuse, highly contextual usage, numerous item types, and

[1] http://www.amazon.com/
[2] http://www.apple.com/itunes/
[3] http://music.yahoo.com/
[4] http://www.pandora.com/
[5] http://www.last.fm

highly passionate users (Celma and Lamere 2011). In order to efficiently access, discover, and present music content to the final user, techniques for searching, retrieving, and recommending need to be appropriate for music content. There has been some work done in both academia and the industry to provide music recommendation services. Understanding patterns of music listening and consumption can help to perform accurate and satisfying music recommendations. This thesis aims to present the development of music recommendation and discovery methods so far, and identify the issues in evaluation that still require careful consideration and research.

Fig. 1.1: An illustration of recommendation systems.



1.1 Music Recommendation in Academia and Industry

Research into music recommendation did not begin in earnest until 2001 (Celma 2010), with an early exception by Shardanand (1994). Over the last decade, both academics and industry members have done work with music recommendation. Music recommendation is an interdisciplinary research area, which involves the areas of search and filtering, musicology, data mining, machine learning, personalization, social networks, text processing, complex networks, user interaction, information visualization, and signal processing (Celma and Lamere 2011). Traditional music recommender systems use collaborative filtering (Shardanand 1994) or content-based filtering (Logan 2004). Collaborative filtering analyzes historical interactions between users and systems and generates recommendations to the user based on the opinions of a group of like-minded users. Alternatively, content-based filtering analyzes the content of items and recommends items similar to the ones the user liked in the past. Since 2005, the research community has broadened its attention to

include other aspects, such as hybrid approaches (Yoshii et al. 2006; Tiemann et al. 2007; Magno and Sable 2008; Yoshii and Goto 2009), context awareness (Park et al. 2007; Shin et al. 2009; Lee et al. 2010), social tagging (Eck et al. 2007; Symeonidis et al. 2008; Nanopoulos et al. 2010), music networks (Buldú et al. 2007; Fields et al. 2008; McFee and Lanckriet 2009; Seyerlehner et al. 2009), visualization (Goto and Goto 2005; Pampalk and Goto 2006), playlist generation (Knees et al. 2006; Flexer et al. 2008; Fields 2011), and group recommendation (Cho et al. 2007; J.-K. Kim et al. 2010). Celma (2010) systematically expounded the topic of music recommendation and discovery in his dissertation.

In the industry, recommendation systems play an important role in e-Commerce. Popular music recommendation services include *Amazon*'s personalized recommendation lists, personalized Internet radio (e.g., *Last.fm* and *Pandora*), and software applications (e.g., Apple's *iTunes Genius* and Microsoft's *Zune MixView*). Other music recommendation systems include *Rhapsody*[6], *iRate*[7], *MusicStrands*[8], *inDiscover*[9], *Spotify*[10], *Play.me*[11], and *seevl*[12]. Companies such as Echo Nest[13] and BMAT[14] provide specific solutions to assist people to discover, personalize, and filter huge amounts of music content.

For evaluating recommendation systems, Netflix, an online streaming video and DVD rental service, announced an open competition[15] for the best recommendation algorithm, offering a prize of $1,000,000 in 2006. The evaluation metric, root mean square error (RMSE), has been widely used to evaluate recommender systems; however, RMSE is not that useful to evaluate music recommendation systems (Celma and Lamere 2011). An overt focus on accuracy does not take into account users' desire in looking for diverse and serendipitous music. Moreover, accuracy metrics like RMSE can only be measured on the test data that has already been rated by users, which may lead to skewness in evaluation. So far, not much work has been done in evaluating how well the music recommenders work. There was a lack of objective evaluation methods and standardized datasets for compar-

[6] http://www.rhapsody.com/start
[7] http://irate.sourceforge.net/
[8] http://strands.com/
[9] http://www.indiscover.net/
[10] http://www.spotify.com/int/
[11] http://www.playme.it/
[12] http://seevl.net/
[13] http://echonest.com
[14] http://bmat.com
[15] http://www.netflixprize.com/

ison (Celma and Lamere 2011). To fill this gap, the *Million Song Dataset Challenge*[16] was released in 2012, which aims at being the best possible offline evaluation of a music recommendation system. It contains listening history for more than 1M users. To predict the missing listening history in the test set, "any type of algorithm can be used: collaborative filtering, content-based methods, web crawling, even human oracles". As a result, this competition will shape the platform for the evaluation of music recommendation systems in the future. More details will be described later in this thesis, in order to speculate on possible methods to evaluate a music recommendation system.

1.2 Music Listening and Consumption

The goal of a music recommendation system is to help consumers and the music industry with the discovery and delivery of music. In order to realize the personalized distribution of music, it may be beneficial for recommender designers to understand the music listening behaviors and know about the state of music consumption in the industry.

Understanding user preference and behavior can help to propose a reasonable recommendation to a specific user. For example, some users show a clear bias towards style when choosing music, while some emphasize timbral similarity (Baumann et al. 2004). In order to make recommendations respectively to these two types of listeners, the recommender needs to focus on different attributes (e.g., musical style and timbre). Moreover, users' feelings and expressions can be different towards the same music (Kodama et al. 2005), such that a personalized user profile is needed for each user before the system can make meaningful recommendations. Generally, a user's preference shifts with time, in terms of years, seasons, days, and even hours (Hayes 2003; Yoon et al. 2008; Park and Kahng 2010; Hu and Ogihara 2011). For instance, a user who liked calm and soft music before, may like noisy music now. So a user's profile needs update and maintenance to describe the music preference of the user at a time. Unlike the consumption of movie, books, and games, people listen to music repeatedly and continuously (Celma and Lamere 2011; Celma 2010; Park and Kahng 2010). This adds more complexity to capture a user's preference accurately, which is important for a music recommendation system.

In reality, the frequency distribution of music transactions is concentrated at the beginning (high volume); dominated by a very few popular items (the well-known hits) followed

[16] http://www.kaggle.com/c/msdchallenge

by a long list of items that does not sell that well (Anderson 2004; Elberse 2007; Celma 2010). This long list is referred as the Long Tail (Anderson 2004), where a large number of songs or artists, the misses, are only played or downloaded by a relatively small group of people. It seems that a majority of users prefer popular titles, while only a minority exploit niche titles. Analyses of the music sales have shown that the music industry is dominated by popular artists and songs (Elberse 2007; Soundscan 2007). However, nearly everybody's taste deviates from the mainstream somewhere, with most people consuming niche products at least some of the time (Anderson 2004; Elberse 2007; Goel et al. 2010). Sometimes, listeners may be expecting to discover and enjoy a wide range of music that may be less popular but a good match to their personal taste (Levy and Bosteels 2010). On the other hand, some economists explored the utility of the tail items and proposed the Long Tail business strategy. That is, offering customization to individual consumers can increase the profit in e-commerce by "selling less of more" (Anderson 2004; Elberse 2007). Both the users and the profit-driven service providers are looking forward to advanced tools for music discovery and recommendation.

1.3 Outline

This thesis identifies the issues that should be addressed to develop music recommendation systems and the ways in which they have been dealt with so far. By examining the historical development of music recommendation methods, the thesis allows designers and researchers to make use of knowledge and experience that have already been accumulated, and to address many open questions that require further exploration. The rest of this thesis is organized as follows:

Chapter 2 describes how collaborative filtering works and reviews its application in music recommendation systems. Collaborative filtering depends on a set of human judgements (known as ratings) for items to predict how well a user will like an item that has not been rated. The rating data can be collected explicitly or implicitly. Memory-based and model-based methods have their own advantages and disadvantages. Although collaborative filtering is a successful recommendation method, it presents some challenges such as data sparsity, popularity bias, and cold start.

Chapter 3 depicts content-based filtering for music recommendation systems. To predict which items the user may like, content-based filtering provides recommendations by

comparing the item information to the user preference. The musical content can be categorized into three types of metadata: editorial, acoustic, and cultural. Given the user preference, the recommendation problem can be treated as an information retrieval task or as a classification task. The content-based filtering can be limited by the content analysis schemes and it has to deal with problems such as scalability and overspecialization.

Chapter 4 presents some new research interests to extend the capabilities of traditional music recommendation systems, including hybrid approaches, context awareness, social tagging, music networks, visualization, playlist generation, and group recommendation.

Chapter 5 investigates how to evaluate a music recommendation system. The evaluation procedure can take into account the properties of music recommendation systems such as accuracy, user preference, coverage, confidence, novelty and serendipity, diversity, and privacy. These properties can be measured in an offline simulation or a user study.

Chapter 6 summarizes the development of music recommendation systems presented in previous chapters and gives suggestions for further work in both the design and the evaluation of music recommendation systems.

Chapter 2

Collaborative Filtering

In everyday life, people rely on others to receive useful recommendations (Resnick and Varian 1997). Collaborative filtering, also termed as social information filtering (Shardanand 1994), is modeled on similar behavior of people to assist and augment automated recommendation systems. It assumes that users who rate items similarly or have similar behaviors (e.g., buying or listening) will rate or act on other items similarly (Sinha and Swearingen 2002). Given a set of human judgements (known as ratings) for items, collaborative filtering aims to predict how well a user will like an item that has not been rated (Herlocker et al. 1999).

The term collaborative filtering (CF) was coined by Goldberg et al. (1992), who developed an e-mail filtering system *Tapestry*. *Tapestry* was expected to scan all mailing lists and select interesting documents for the user. However, the collaborative filtering provided by *Tapestry* was not automated and required users to construct complex queries in a special query language designed for the task (Herlocker et al. 1999). Later, *GroupLens* (Resnick et al. 1994) introduced an automated collaborative filtering system to help people find articles they may like in the huge stream of available articles. Other early implementations of collaborative systems were published with projects such as personalized music recommender *Ringo* (Shardanand 1994; Shardanand and Maes 1995), video recommender (Hill et al. 1995), and *MovieLens* (Dahlen et al. 1998). Today, automated collaborative filtering has been widely used in both information filtering applications and E-commerce applications (Sarwar et al. 2001; Linden et al. 2003). It has been incorporated by Inter-

net retailers (e.g., Amazon, Barnes and Noble[1]), music recommenders (e.g., *MyLaunch*[2], *Last.fm*), movie recommenders (e.g., *Internet Movie Database*[3], *Rotten Tomatoes*[4]), and so forth.

Collaborative filtering depends on a database of preferences for items by users to predict a user's affinity for items or the utility of items to a user (Breese et al. 1998). Memory-based CF methods use the user rating data to calculate the similarity between users or items and make recommendations according to those similarity values. On the other hand, Model-based CF methods use the rating data to learn a model to make predictions. Each method has advantages and disadvantages in implementing a recommendation system. This chapter will describe how collaborative filtering works and discuss the challenges it has to address for a music recommendation system.

2.1 User-Item dataset

In a typical collaborative filtering scenario, the database is a collection of interactions between m users and n items. Given a list of m users $U = \{u_1, u_2, ..., u_m\}$ and a list of n items $I = \{i_1, i_2, ..., i_n\}$, the dataset is generally represented as a user-item matrix R. In matrix R, each row represents a user profile, whereas each column corresponds to one item (Celma 2010). The value $r_{u,i}$ is the rating of the user u for the item i, as depicted in Fig. **??**.

A common approach to build such a user preference dataset is through eliciting either explicit ratings or implicit inferences from users, which provide different degrees of expressivity of users' preferences (Jawaheer et al. 2010). Explicitly, user can choose to rate items with an numerical scale, e.g., 1–5. Amatriain et al. (2009) showed that explicit feedback suffered from natural noise that referred to user variability and inconsistency in providing explicit feedback. It is beneficial to exploit which perspective had been selected by the user to rate music, such as lyrics, melodies, instruments, singers, originality, and impression (Anderson et al. 2003; Chedrawy and Abidi 2006). As an alternative to explicit ratings, implicit inference can be generated by analyzing user's behavior, which is domain-dependent (Jawaheer et al. 2010). For example, website recommenders can use a

[1] http://www.barnesandnoble.com/
[2] http://www.mylaunch.com
[3] http://www.imdb.com
[4] http://www.rottentomatoes.com

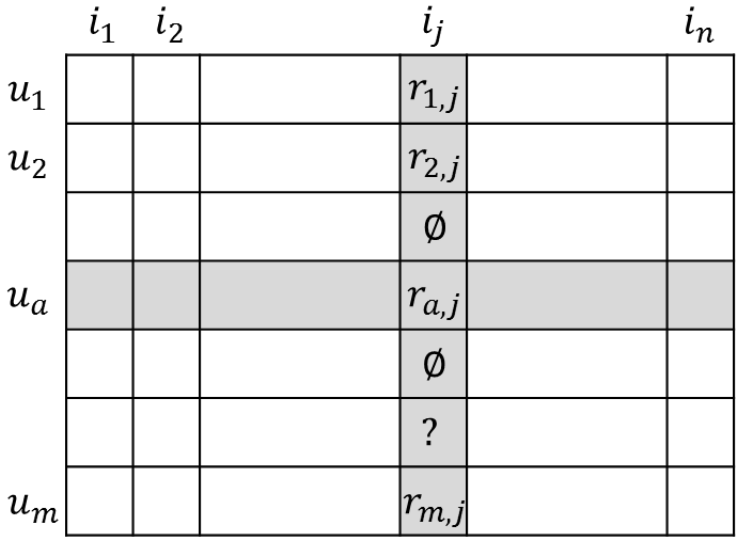


Fig. 2.1: User-item matrix for collaborative filtering.

record of browsed URLs, while book recommenders can employ purchase history. In music recommendation, tracking user listening habits (e.g., in terms of playcounts) has become the most common way (Celma 2010)

According to Loeb (1992), implicit feedback is important for less interactive or casual users, for they are not likely to be willing to engage in lengthy interactions with the system. However, some researchers pointed out that it could only capture positive preferences (Jawaheer et al. 2010; Celma 2010). For example, if a user skipped a recommended track, that does not imply the user does not like the track. It may be because that track doesn't match the current emotion or context. On the other hand, explicit feedback can be both positive and negative. The rating scale can go from 'I like it a lot' to 'I do not like it at all'. Explicit feedback tend to be absolute whereas implicit feedback tend to be relative (Jawaheer et al. 2010). For example, a user, listening to track A ten times, may express high preference if the user typically listens to each track once or twice. Or the user's preference on track A may be low if he/she listens to other tracks more than 20 times.

Explicit and implicit feedback exhibits different characteristics to represent users' preferences with both pros and cons. Some researchers have exploited new techniques to use explicit or implicit feedback. RACOFI (rule applying collaborative filtering approach) (Anderson et al. 2003) incorporated a set of logic rules, which can implicitly modify the ratings that a user has done previously. For example, "if a user rates 9 the originality of an album

by artist X then the predicted originality rating, for this user, of all other albums by artist X is increased by a value of 0.5". Hu et al. (2008) proposed the notion of applying confidence levels to interpret the implicit feedback as positive and negative preference values.

2.2 Memory-based CF Techniques

Memory-based CF algorithms work over the entire or a sample of the user-item database to predict the utility of items to the active user (the user under current consideration for recommendations) (Breese et al. 1998; Sarwar et al. 2001; Melville and Sindhwani 2010). Generally, they compute similarity between users or items, and then use the weighted average of ratings to produce predictions based on the similarity values (Su and Khoshgoftaar 2009). According to the similarity employed, either among users or among items, the task of prediction can be done in user-based neighborhood or item-based neighborhood. Making use of these neighborhood-based methods, top-N recommendation generates a list of N top-ranked items that will be of interest to the active user (Karypis 2001).

2.2.1 User-based Neighborhood

User-based method calculates the similarity between the users u and v based on the items they have both rated (Su and Khoshgoftaar 2009). To compute the similarity between two users, Pearson correlation (Shardanand 1994; Breese et al. 1998; Hayes 2003; Sarwar et al. 2000) and cosine similarity (Breese et al. 1998; Sarwar et al. 2000) are often utilized. Let I denote the set of items that have been rated by users u and v, Pearson correlation will be

$$sim(u,v) = \frac{\sum_{i\in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i\in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i\in I}(r_{v,i} - \bar{r}_v)^2}}, \tag{2.1}$$

where $\bar{r}_u$ is the average rating of user u. Constrained Pearson correlation (Shardanand 1994) is a variation of Pearson correlation, which uses midpoint instead of mean rate and is given by,

$$sim(u,v) = \frac{\sum_{i\in I}(r_{u,i} - r_m)(r_{v,i} - r_m)}{\sqrt{\sum_{i\in I}(r_{u,i} - r_m)^2}\sqrt{\sum_{i\in I}(r_{v,i} - r_m)^2}}, \tag{2.2}$$

where r_m is the mid-point of rating scale.

Vector cosine similarity is defined as,

$$sim(u,v) = \frac{\sum_{i \in I} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I} r_{u,i}^2} \sqrt{\sum_{i \in I} r_{v,i}^2}}. \tag{2.3}$$

It is possible to use mean squared difference (MSD) (Shardanand 1994) to derive the similarity between a pair of users. MSD is formulated as

$$MSD(u,v) = \frac{\sum_{i \in I} c_{u,i} c_{v,i} (r_{u,i} - r_{v,i})^2}{\sum_{i \in I} c_{u,i} c_{v,i}}, \tag{2.4}$$

where $c_{u,i} = [1, 0]$ if $r_{u,i}$ [is, is not] defined.

Obtaining the predicted preference of a user on an item is the most important step in a collaborative filtering system (Su and Khoshgoftaar 2009). For the user-based method, a subset of nearest neighbors of the active user is chosen based on their similarities, and the prediction is generated as a weighted aggregate of their ratings (Herlocker et al. 1999; Celma 2010). Using the similarity as weighting, the predicted value for item i to user u is

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{v \in S^k(u)} sim(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{v \in S^k(u)} sim(u,v)}, \tag{2.5}$$

where $\bar{r}_u$ is the average rating of user u, and $S^k(u)$ denotes the set of k neighbors for user u.

User-based approaches by dynamically computing a neighborhood of similar users are better suited to provide truly personalized information (Karypis 2001). However, they have limitations related to scalability and real-time performance (Linden et al. 2003; Deshpande and Karypis 2004; Sarwar et al. 2001). The bottleneck is the search for neighbors among a large user population of potential neighbors (Herlocker et al. 1999). The computational complexity grows linearly with the total number of users. Moreover, this type of algorithm cannot take advantages of pre-computed user-to-user similarities (Karypis 2001). Without the aid of pre-computation, the latency is crucial for near real-time performance (Sarwar et al. 2001; Deshpande and Karypis 2004).

2.2.2 Item-based Neighborhood

Sarwar et al. (2001) proposed using the same correlation-based and cosine-based techniques to compute similarities between items instead of between users. This idea has been further

extended in Deshpande and Karypis (2004) for the top-N item recommendations. To obtain the similarity between items i and j, the algorithms work on the set of users U, who have rated them both. The common approaches to calculate the item similarity are Pearson correlation (Eq. 2.1) and cosine similarity (Eq. 2.3).

Considering the reality that different users may use different rating scales, researchers exploit adjusted cosine similarity (Deshpande and Karypis 2004; Sarwar et al. 2001) to compute the similarity between two items, which is

$$sim(i,j) = \frac{\sum_{u \in U}(r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{u \in U}(r_{u,j} - \bar{r}_u)^2}}, \tag{2.6}$$

where $\bar{r}_u$ is the average rating of user u.

Similarity using conditional probability (Karypis 2001) is defined as

$$sim(i,j) = P(j|i) \simeq \frac{f(i \cap j)}{f(i)}, \tag{2.7}$$

where $f(I)$ is the number of users who purchased item I. Note that this metric is asymmetric.

Taking advantage of the set of k nearest neighbors of item i, $S^k(i;u)$, the item-based methods generate the predicted utility as a simple weighted average of all ratings from user u (Celma 2010; Sarwar et al. 2001).

$$\hat{r}_{u,i} = \frac{\sum_{j \in S^k(i;u)} sim(i,j) r_{u,j}}{\sum_{j \in S^k(i;u)} sim(i,j)} \tag{2.8}$$

The assumption behind the item-based scheme is that a user will be interested in purchasing items that are similar to the items the user liked earlier and will tend to avoid items that are similar to the items the user disliked earlier (Sarwar et al. 2001; Deshpande and Karypis 2004). The item-based algorithms are expected to alleviate the scalability problem of the user-based algorithms and lead to a much faster recommendation engine. The computationally expensive part is to build an item-to-item matrix of similarities between each pair of items, which can be done offline. The online task is to select items similar to each of the active user's purchases, aggregate them, and generate the recommendation. Thus, the online computation is independent of the total number of items or users, but only dependent on the number of items that have been rated or purchased by the active user. This characteristic guarantees the computation speed for even extremely large datasets (Linden

et al. 2003). On the other side, the item-to-item recommendation tends to generate lower-quality recommendations than the user-based method (Sarwar et al. 2001). And another potential limitation is that the globally computed item-to-item similarities may not be able to provide a sufficient degree of personalization (Karypis 2001).

2.3 Model-based CF Techniques

The Model-based CF algorithms use the database to learn or build a descriptive model of users, items, and/or ratings, which is then applied to make predictions (Breese et al. 1998; Pennock et al. 2000). From a probabilistic perspective, the collaborative filtering task can be viewed as calculating the expected value of a rating, $\hat{r}_{u,i}$, given the rating records of the active user u (Breese et al. 1998). If K denotes the set of rating values and I_u denotes the set of items on which user u has rated, the prediction is given by

$$\hat{r}_{u,i} = \sum_{k=0}^{K} P(r_{u,i} = k | r_{u,j}, j \in I_u)k, \tag{2.9}$$

where $P(r_{u,i} = k|r_{u,j}, j \in I_u)$ is the probability that the active user u will have a particular rating for item i given the previously observed ratings. These probabilities are the parameters of the desired model, which are estimated from a training set of user ratings via offline computation. The model building process can be performed by many machine learning and data mining algorithms such as clustering (Breese et al. 1998), regression (Sarwar et al. 2001), and other models.

2.3.1 Clustering Model

Clustering models can explicitly infer the preferences of the underlying users (Su and Khoshgoftaar 2009). In Breese et al. (1998), the clustering model was used to treat collaborative filtering as a classification problem. Using a Bayesian classifier, it clustered similar users into the same class, estimated the probability that a particular user was in a particular class, and from there computed the conditional probability of ratings. Based on the concept of classification, Ungar and Foster (1998) clustered users and items separately using variations of k-means and *Gibbs* sampling. Each user was assigned to a class with a degree of membership proportional to the similarity between the user and the mean of the class.

Some researchers also applied a data clustering algorithm as an intermediate step. These clustering methods (O'Connor et al. 2002; Anglade et al. 2007b) required a distance metric or similarity metric to guide the partitioning process, grouped users or items, and then performed recommendation within each cluster independent of the other clusters. O'Connor et al. (2002) performed partitioning in the item space and applied traditional collaborative filtering in each cluster of items. Anglade et al. (2007b) presented a peer-to-peer music recommendation system that clustered users with similar tastes and provided shared music radio channels to them.

2.3.2 Regression Model

In practice, two rating vectors may be distant in terms of Euclidean distance but are highly similar with respect to Pearson correlation or vector cosine measure (Su and Khoshgoftaar 2009). In this case, using the raw ratings of the so-called similar items may result in poor prediction. Instead of using raw ratings, a regression model uses an approximation of the ratings (Sarwar et al. 2001; Su and Khoshgoftaar 2009). Given the user-item matrix $R_{m\times n}$, the linear regression model can be expressed as

$$R = MX + N, \tag{2.10}$$

where M is a $m \times k$ matrix with each row as an estimate for one user, X is a $k \times n$ matrix, and N is a random variable representing noise in user choices.

Vucetic and Obradovic (2005) proposed a regression-based CF approach that first learned a number of experts describing relationships in ratings between pairs of items and then combined the experts using statistical methods to predict the active user's preferences. They used ordinary least squares to estimate the parameters of the linear regression function. Lemire and Maclachlan (2005) proposed three Slope One schemes with the predictors of the form $f(x) = x + b$, which precomputed the average difference between the ratings of one item and another for users who rated both. The assumption behind is that the popularity differential between items for users can determine how much better one item is liked than another on average. In return, the average popularity differential between a pair of items can be used to predict a user's preference on one unrated item if the user has rated another.

2.3.3 Other Models

In the literature, researchers have also incorporated other models into collaborative filtering systems. Breese et al. (1998) formulated a probabilistic collaborative filtering with the Bayesian network model, where a node corresponded to an item and the state of each node corresponded to possible rating values for each item. The learning algorithm searched over various model structures to capture dependencies for each item and the resulted model had a decision tree at each node to encode the conditional probabilities.

An alternative to Bayesian network is a dependency network (Heckerman et al. 2000), a graphical model for probabilistic relationships. Similar to a Bayesian network, a dependency network is a directed graph, where a node's parents render that node with a conditional probability. The dependency network is potentially cyclic while the Bayesian network is acyclic. Although less accurate than Bayesian networks, dependency networks are faster in generating predictions and require less time and memory to learn.

Sarwar et al. (2000) applied association rule discovery algorithms to develop top-N recommendation systems. Given two subsets of items A and B, such that $A, B \subseteq I$ and $A \bigcap B = \emptyset$, an association rule states that the presence of items from A in a transaction indicates a likelihood that items from B are also present in the transaction. The confidence of an association rule can be determined by the fraction of transactions that involve both A and B. This rule-based model discovered association between co-purchased items and then generated item recommendations based on the strength of the association between items.

Billsus and Pazzani (1998) proposed a machine learning framework for collaborative filtering, where various machine learning techniques (such as artificial neural networks) coupled with dimensionality reduction techniques (such as singular value decomposition) can be used. Other model-based collaborative filtering methods include a probabilistic relational model (Getoor and Sahami 1999), a maximum entropy model (Pavlov and Pennock 2002), a probabilistic latent semantic analysis model (Hofmann 2004), and a Markov decision process model (Shani et al. 2002).

2.4 Hybrid Memory- and Model-based Techniques

Memory-based CF algorithms are easy to implement, seem to work well with dense datasets in practice, and are able to add new data easily and incrementally. However, this approach can become computationally expensive, in terms of both time and space, as the size of the

database grows. Moreover, data sparsity may decrease the performance of recommendation, and these methods generally cannot provide explanations of recommendations to users (Deerwester et al. 1990; Pennock et al. 2000; Su and Khoshgoftaar 2009).

Model-based CF algorithms can alleviate the sparsity and scalability problem to some extent. Beyond the predictive capabilities, the model can highlight certain correlations in the data, offer an intuitive rationale for recommendations, or make assumptions more explicit. The generated model can calculate predictions quickly and the memory requirements for the model are generally less than for storing the full database (Pennock et al. 2000). On the other hand, the time complexity to build a model may be prohibitive, and adding one new data entry may require a full recompilation. When the dataset becomes extremely sparse, model-based algorithms may not be practical, however, and for many algorithms, there is a tradeoff between prediction performance and scalability (Su and Khoshgoftaar 2009).

It is possible to combine the memory-based and model-based CF algorithms into a hybrid approach. Personality diagnosis (Pennock et al. 2000) is a representative hybrid CF approach that retains some advantages of both algorithms. Like memory-based methods, the hybrid approach maintains all data throughout the process, and does not require a compilation step to incorporate new data. This approach manifests each user's underlying personality type with their ratings for items, and assumes that users report ratings with Gaussian noise. Given a user's known ratings, it computes the probability that the user is of the same personality type as other users, and then predicts the probability that the user will like new items. The empirical results demonstrated that the use of this combined approach can provide better recommendations than pure memory-based and model-based collaborative filtering. The approach of personality diagnosis retains some of the advantages of both memory- and model-based algorithms, namely simplicity, extensibility, normative grounding, and explanatory power. ClustKNN (Rashid et al. 2006) is a highly scalable hybrid algorithm. This method first compresses data tremendously by building a straightforward but efficient clustering model, and then generates recommendations quickly by using a simple nearest neighbor-based approach.

2.5 Collaborative Music Recommendation Systems

The development of collaborative music recommendation systems can be traced back to the 1990s. *Ringo* (Shardanand 1994; Shardanand and Maes 1995) is one of the first generation of music recommendation systems. It collected user ratings via email and made personalized recommendations for music albums and artists. As a memory-based CF system, *Ringo* utilized three functions (mean square difference, Pearson correlation, and constrained Pearson correlation) to calculate the similarities between users and employed Pearson correlation to implement the artist-to-artist recommendation.

The *Audiomomma* music recommendation system (Alvira et al. 2001) incorporated a memory-based CF approach. It kept track of what artists a user listens to, searched for other users with similar tastes based on the user's listening history, and recommended other artists that these similar listeners enjoyed.

A website-oriented music recommendation system (Chen and Chen 2001, 2005) provided the service of music recommendation based on music grouping and user grouping. The music grouping was achieved by analyzing the content of music items. By recording a user's access histories (i.e., the access time, the music item id, the corresponding music group which the item belongs to, and the corresponding transaction), the system derived an interest profile and a behavior profile of the user. Instead of predicting the utility of one item to the active user, the user-based method predicted the association of each music group with the active user.

Smart Radio (Hayes 2003), a community-based music radio, was a web-based application that allowed its users to manage and share their personalized music programmes. It was "a dynamic domain where users may seek new recommendations almost immediately once the current programme has played through". Thus, for the concern of computation and scalability, a scheduler recalculated the neighborhood for each online user every half hour for the recommendation engine to perform a memory-based CF algorithm.

As mentioned in Celma (2010), *inDiscover* implements the Slope One collaborative filtering method (Lemire and Maclachlan 2005) as well as the RACOFI (rule applying collaborative filtering) approach (Anderson et al. 2003). RACOFI incorporates a set of logic rules with the goal to promote the items that the user will be most familiar with. As an example, "if a user rates 9 the originality of an album by artist X then the predicted

originality rating, for this user, of all other albums by artist X is increased by a value of 0.5". In this way, the ratings have been implicitly modified.

Last.fm is a music recommendation site and a streaming radio station with a built-in collaborative filter that attempts to learn its listeners' likes and dislikes. For each user, *Last.fm* builds a preference profile either by means of the *Scrobbler* or by recording "Skipping" behaviors. With the *Scrobbler*, *Last.fm* can be informed of which songs the user listens to on *iTunes*, *Spotify*, and various other web services and software. Alternatively during the music streaming, if a song plays to the end, it is recorded as like. But if the user skips a song, it is recorded as dislike. Over time, a preference profile is built. Based on which artists a user listens to, *Last.fm* can find musical neighbors who listen to similar music, and explore their collections to recommend new music to the user.

Genius, a playlist generation feature of *iTunes*, make recommendations depending on the collaborative wisdom of crowds (Barrington et al. 2009; Mims 2010; Bogdanov and Herrera 2011). Mims (2010) hints at how *Genius* may work according to the factors defined by the algorithms. Basically, it compares the seed song to *iTunes*' massive database of music, based on song rating data, play history, and metadata. When it is first initialized, *Genius* analyzes a user's music library and compiles all of the data necessary to build playlists (Barrington et al. 2009).

2.6 Challenges

Collaborative filtering is a subjective method that aggregates the social behavior of the users to make recommendations (Celma 2010). It is based on the user preference dataset (in terms of ratings, purchases, downloads, etc.), and does not take into account the description of the items. It presents some challenges although it is a popular recommendation method.

2.6.1 Data Sparsity

Data sparsity is an inherent property of user preference datasets since users only rate the items they have accessed or purchased. Those commercial systems with a relative large number of users and items have the problem of low coverage of the users' ratings among the items. It is common to have a sparse user-item matrix of 1% (or less) coverage (Celma 2010). The lack of access to the content of the items prevents similar users from being matched unless they have rated the exact same items (Balabanović and Shoham 1997). Thus, the

sparsity of dataset prohibits effective recommendations. Only the items that were actually rated by users can be addressed and only strong patterns in communities are actually propagated (Pachet 2003). If two similar items have never been rated by the same user, those two items cannot be classified into the same community, not necessarily because they are inherently unassociated but because their associations have not been observed among the users (Li et al. 2007). For recommendation systems that rely on comparing users in pairs and therefore generating predictions, data sparsity poses challenges to neighbor transitivity (Sarwar et al. 2001). If user A and B have similar interests, and user C shares similar interests with B, it is not necessarily true that user A and C are like-minded as they may have rated too few items.

To alleviate the data sparsity problem, many approaches have been proposed, including Singular Value Decomposition (SVD) (Billsus and Pazzani 1998), Principal Component Analysis (PCA) (Goldberg et al. 2001), and Latent Semantic Indexing (LSI) (Deerwester et al. 1990). Generally, model-based CF algorithms can tackle the sparsity problem better than memory-based CF algorithms (Sarwar et al. 2001; Su and Khoshgoftaar 2009). First, it is easier to capture the similarities between users and items in a reduced dimensional space than in a sparse high-dimensional space (Sarwar et al. 2001). Second, model-based algorithms can provide more accurate predictions for sparse data than memory-based ones (Su and Khoshgoftaar 2009).

2.6.2 Scalability

Traditional collaborative filtering will suffer scalability problems when the number of users and items grow tremendously. Many systems need to react immediately to online requirements and make recommendations for all users (Linden et al. 2003). A poorly scalable system will slow this process. Model-based CF algorithms, such as clustering CF algorithms, address the scalability problem by seeking candidates for recommendation within smaller and highly similar clusters instead of the entire database (O'Connor et al. 2002). But for those models there are tradeoffs between scalability and quality. Dimensionality reduction techniques can tackle the scalability problem and still produce accurate recommendations, but they have to undergo expensive matrix factorization steps (Su and Khoshgoftaar 2009).

2.6.3 Popularity Bias

Collaborative filtering is prone to popularity bias as expected by its inherent social component. It "tends to reinforce popular artists, at the expense of discarding less-known music" (Celma and Cano 2008). The popularity of music can be measured in terms of total playcounts (Celma and Cano 2008) or the fraction of total consumption fulfilled (Goel et al. 2010). In a user preference dataset, popular items seem to be similar to (or related with) lots of items, such that they are more likely to be recommended. As a consequence, the recommenders are sometimes biased towards a small number of popular items and do not explore the Long Tail of unknown items that could be more interesting and novel for the users (Celma 2010). Navigation through the network of popular artists reveals a poor discovery ratio. And this can decrease user satisfaction and novelty detection in the recommendation workflow (Herlocker et al. 2004; McNee et al. 2006). On the other hand, content-based and human expert-based recommendation systems are unvulnerable to the popularity bias. One possible way to recommend long tail items using conventional collaborative filtering, is to identify a candidate pool of long tail items from which to draw recommendations (Levy and Bosteels 2010).

2.6.4 Cold Start

The *cold start* problem is related to both elements of a recommendation system: users and items. It occurs when a new user or item has just entered the system, which is also called *new user/item* problem (Adomavicius and Tuzhilin 2005) or *early* rater problem (Claypool et al. 1999). It is difficult to find similar users or items because there is not enough information. Collaborative filtering cannot recommend a new item until some users rate it, for there are no user ratings on which to base the predictions. Moreover, early recommendations for the item will often be inaccurate because there are few ratings on it. Similarly, new users are unlikely to receive good recommendations because of the lack of their ratings or purchase histories. The *cold start* problem essentially restricts the performance of a collaborative filtering system (Celma 2010).

2.6.5 Shilling Attacks

Since collaborative filtering relies on social information to receive recommendations, good ratings seem to promise a good selling rate. To manipulate the recommendation, producers

or malicious users may introduce fake user profiles that highly rate a set of target items, and then give negative ratings to other items. The desired result is known as a *shilling attack*, which consists of either increasing rating (push attack) or lowering rating (nuke attack) (Chirita et al. 2005). These attacks can affect the quality of the recommendation and result in decreasing satisfaction with the system. Lam and Riedl (2004) found that item-based CF algorithms were much less affected by the attacks than the user-based CF algorithms. Attack models for shilling the item-based CF systems have been examined in Mobasher et al. (2005). Future systems need to introduce precautions that discourage this phenomenon (Resnick and Varian 1997).

2.6.6 Gray Sheep

The users, whose opinions do not consistently agree or disagree with any group of people, will rarely receive accurate collaborative filtering predictions, even after the initial start up phase. This problem is referred as *Grey sheep* (Claypool et al. 1999). Their atypical tastes makes it difficult to find many users as their neighbors (Celma 2010), and thus these users will not benefit from pure collaborative filtering systems.

2.6.7 Explainability

According to Herlocker et al. (2000) and Sinha and Swearingen (2002), a good recommendation system should be able to explain to the user why the particular item has been recommended. The explanations can help users justify how much confidence to put in the received recommendation. Therefore, the explainability of a recommendation system can increase the user acceptance by providing a justification of the system's choice beyond a blind recommendation. This is very beneficial when the CF-based recommendations are not satisfying because users' trust in the system may be degraded by these inapposite recommendations. A user study of five music recommendation systems, conducted by Sinha and Swearingen (2002), indicated that users felt more confident in recommendations perceived as explainable and more inclinable to accept them. Herlocker et al. (2000) explored the effective models in supporting explanations in CF systems and believed that providing explanations could also increase the filtering performance of CF systems.

Chapter 3

Content-based Filtering

Content-based filtering provides recommendations by comparing the item (e.g., music) information to the user preference, with the goal of predicting which items the user may like (Celma 2010; Melville and Sindhwani 2010). Unlike collaborative filtering based on other users' ratings, content-based filtering focuses on the properties of items. The main assumption under such approaches is that an item can be identified by a set of features extracted directly from their content (Orio 2006). Content-based recommendation systems attempt to make suggestions based on the analysis of items or the metadata associated with them.

The content-based recommendation approach has its roots in the information retrieval (IR) field (Adomavicius and Tuzhilin 2005; Celma 2010). Many early content-based systems focused on recommending items containing textual information, such as documents and Web sites (URLs) (Adomavicius and Tuzhilin 2005; Melville and Sindhwani 2010). Recently, this approach has been applied to the multimedia domain. A popular system using (manual) content-based descriptions to recommend music is Pandora (Celma 2006), which plays music with similar characteristics to those of a song provided by the user as an initial seed. Pandora employs a team of professional music analysts to collect musical details on every track, such as melody, harmony, instrumentation, rhythm, and vocals.

The general idea of content-based music recommendation systems is to analyze the content of music items and decide the proper suggestions based on the comparison to user preferences. The description of musical content can be collected either manually (like Pandora Radio) or automatically. The manual description process is expensive in terms of both labor and time (Magno and Sable 2008; Celma 2010). A lot of research work

have focused on automatic acoustic feature extraction or web mining techniques to access cultural information (Celma 2010). The user preference can be either represented by a seed song or derived from listening or consumption histories. The process of matching candidate items against user preference, can be treated as an information retrieval task or a classification task (Melville and Sindhwani 2010).

3.1 Music Content Analysis

Content analysis is to create an item profile for representation. An artist or an album can be recognized by various tags or the aggregation over all songs labeled with that artist. The content of a given song can be obtained through an acoustic feature detection approach (Huang and Jenor 2004; Li et al. 2004; Logan 2004; Knees et al. 2006; Shao et al. 2009), a metadata (e.g., genre, artists, and lyrics) mining approach (Aucouturier and Pachet 2002c; Pauws et al. 2006; Platt et al. 2002; Ragno et al. 2005), or the analysis of a score-based symbolic representation (Chen and Chen 2005; Vembu and Baumann 2005). Pachet (2005) proposed a framework to categorize music information into editorial, acoustic, and cultural metadata, such that music content analysis is to retrieve the three types of metadata.

3.1.1 Editorial Metadata

Editorial metadata refers to the metadata manually entered by an editor, usually an expert or a group of experts (Pachet 2005). It covers anything from album information (e.g., the song "I Feel Good" by James Brown was released as a single in 1965) to administrative information such as artist biography, genre information, and relationships among artists. Thus, editorial metadata is not necessarily objective. A well-known music database organized by genre is Allmusic[1], where professional editors compose brief descriptions of popular artists and often include a list of similar artists (Berenzweig et al. 2004). *mpME!* (Dunne et al. 2002) recommended music or artist to its users by using the artist directories on AllMusic and Ultimate Band List (UBL)[2].

[1] http://www.allmusic.com/
[2] http://www.ubl.com

3.1.2 Acoustic Metadata

Acoustic metadata is obtained by analyzing audio files without any reference to a textual or prescribed information. Thus, acoustic metadata is purely objective information, pertaining to the content of music (Pachet 2005). Acoustic metadata includes low-level signal features (e.g., temporal features and spectral features) and high-level musical concepts (e.g., rhythm, melody, and timbre) (Baumann and Hummel 2005; Casey et al. 2008; Celma 2010). One noticeable endeavor promoting automatic feature extraction is the music information retrieval experimental exchange (MIREX). MIREX provides a framework for formal evaluation of music information retrieval (MIR) systems using centralized tasks, datasets, platforms, and evaluation methods (Casey et al. 2008). Through machine learning algorithms such as regression, the acoustic metadata can be used to infer semantic descriptors, such as genre, mood, and instrumentation (Bogdanov et al. 2010).

Low-level Signal Features

Low-level audio features are measurements of audio signals that contain information about a musical work (Casey et al. 2008). According to the domain of analysis (time or frequency), these features are generally categorized into temporal features or spectral features . Most of the time, the signal is segmented into (overlapping) frames from 10 to 100 ms with 50% overlap (Chen and Chen 2005; Kodama et al. 2005; Pampalk et al. 2005; Celma 2010). For each frame, a feature vector is computed. Considering the large number of frames, the resulted representation of a whole music piece can be characterized with one averaged feature vector (Magno and Sable 2008) or one model (Cai et al. 2007).

Temporal Features are extracted directly from the raw audio signal without any preceding transformations, such that the computational complexity of temporal features tend to be low (Mitrović et al. 2010). Zero crossing rate (ZCR) is such a inexpensive and simple feature that is defined as the number of zero crossing in the temporal domain within a time frame. It may be used to provide a measure of the noisiness of the signal (D.-M. Kim et al. 2007; Yoon et al. 2008), which is related to the timbral texture (Bozzon et al. 2008). For example, heavy metal music tends to have a higher ZCR than classical music due to guitar distortion and drums (Tzanetakis and Cook 2002). The amplitude-based or power-based

features represent the temporal envelope of an audio signal. These features are easy and fast to compute but limited in their expressiveness (Mitrović et al. 2010).

Spectral Features are considered more robust to polyphonic and complex textures (Celma 2010). The popular ways to extract spectral features include Fourier transform, autocorrelation, cosine transform, wavelet transform, and the constant Q transform (Casey et al. 2008). The group of spectral features is usually the largest group of audio features (Mitrović et al. 2010). Here, I focus on the features that have been applied to music recommendation in the literature.

The short-time Fourier transform (STFT) performs a time-frequency decomposition as the computation of spectrogram. STFT yields complex values, which carry both the distribution of frequency components and the phase information on these components. The frequency distribution is an estimation of the magnitude spectrum, which often serves as the first step of feature extraction (Casey et al. 2008; Huang and Jenor 2004). After computing the spectral envelope, many low-level audio features can be obtained, such as spectral rolloff, spectral flatness, and spectral flux. Both the spectral rolloff and the spectral flatness measure the spectral shape and indicates the tonality of sound. Noise-like sounds tend to have a flat or uniform spectrum, while tonal sounds typically have line spectra (Mitrović et al. 2010). The spectral rolloff point is the $N\%$ percentile of the power spectral distribution, where N is usually 85% or 95% (Scheirer and Slaney 1997). In other words, it is the frequency below which $N\%$ of the magnitude distribution is concentrated (Mitrović et al. 2010). Besides music recommendation (D.-M. Kim et al. 2007; Yoon et al. 2008), this measure has also been used in genre classification (Li and Ogihara 2005) and timbre modeling (Morchen et al. 2006). The spectral flatness is defined as the ratio of the geometric and the arithmetic mean of a subband in the power spectrum. It shows to which degree the frequencies in a spectrum are uniformly distributed (Huang and Jenor 2004). Roughly, a flatness close to 1 means a "noise-like" signal, while a value close to 0 means a "tone-like" signal. The spectral flux measures the frame-to-frame spectral amplitude difference, quantifying changes in the shape of the spectrum over time (Scheirer and Slaney 1997; D.-M. Kim et al. 2007). The spectral flux is an important perceptual attribute in the characterization of timbre (Yoon et al. 2008). Fluctuation patterns (Pampalk et al. 2005) characterize the energy distribution by describing the power fluctuations in different

frequency bands (Pampalk et al. (2005) used 20 frequency bands according to the Bark scale).

The Mel-frequency Cepstral Coefficients (MFCCs) represent the spectral envelope of a signal, based on a conversion of the log amplitude spectrum to Mel-scale and a discrete cosine transform (DCT) (Mitrović et al. 2010). MFCCs have been useful for the timbre and genre classification (Celma 2010). In the literature (Logan 2004; Pampalk et al. 2005; Knees et al. 2006; Magno and Sable 2008), the MFCC feature set has already shown effective performance for the task of music recommendation. Foote (1997) utilized the histograms of MFCCs to develop a music indexing system; while others have aggregated MFCCs together. Logan and Salomon (2001, 2002, 2004) modeled a group of MFCC features with K-means clustering, while Aucouturier and Pachet (2002a) and Knees et al. (2006) modeled the distribution of each song's MFCCs as a Gaussian Mixture Model (GMM) over the space defined by MFCC features. Magno and Sable (2008) first performed the K-means clustering on the MFCC set of a target song, and then fit each data cluster with a Gaussian component to subsequently form a GMM.

Linear predictive coding (LPC) aims to estimate the filter of a source-filter model of signal production. The LPC spectrum can be used as an approximation of the spectral envelope. Due to the high retrieval efficiency of LPC coefficients, it has been extensively used in automatic speech recognition and audio retrieval (Mitrović et al. 2010), and has recently been adopted by music classification algorithms (Xu et al. 2005) and music recommendation systems (Dickerson and Ventura 2009)

High-level Music Concepts

High-level music concept embodies the intuitive information that a sophisticated listener would have about a piece of music (Casey et al. 2008). In the literature (Kuo and Shan 2002; Chen and Chen 2005; Cano et al. 2005; Kodama et al. 2005; Hijikata et al. 2006), recommendation systems extracted features of timbre, melody, rhythm, and music structure, based on the computation of low-level signal features.

Timbre is the quality of sound which allows human ears to distinguish the sound that have the same pitch and loudness (Winckel and Binkley 1967; Khine et al. 2008). It is mainly determined by the harmonic content and the dynamic characteristics of the sound such as attack-decay envelope (Winckel and Binkley 1967). In addition to representing

the distinguishable characteristics of a tone (Khine et al. 2008), the timbral features can even help to differentiate mixture of sounds with similar rhythmic content (Çataltepe and Altinel 2007). In this case, the spectral envelope consists of the basis for the description of timbre (Schwarz and Rodet 1999). It is believed by Aucouturier and Pachet (2004) that timbre is often correlated with music taste such that timbre similarity is a natural way to measure relationships between music. According to McKay and Fujinaga (2006), many genre classification systems have utilized primarily timbral features. In the development of content-based music recommendation systems, researchers have used MFCC (Magno and Sable 2008; Chordia et al. 2008), STFT (Çataltepe and Altinel 2007; D.-M. Kim et al. 2007), and modified discrete cosine transform (MDCT) (Zhu et al. 2006) to capture timbral features.

Melody can be a strong identifier of Western music (Casey et al. 2008). Psychoacoustic research (Dowling 1978) has shown that the contour, or shape, of a melody is an important memorable feature for music. According to Downie (2003), any representation that highlights a work's melody can increase the chances of successful music identification or retrieval. For example, the estimated melody line facilitated song retrieval based on similar singing voice timbres (Fujihara and Goto 2007). Melody can be represented as a continuous temporal-trajectory representation of fundamental frequency (pitch) or a series of musical notes (Casey et al. 2008). In this case, the estimation of melodies can be achieved by finding the predominant fundamental frequency trajectory (Goto 2004) or through a knowledge-based (Eggink and Brown 2004) or classification-based (Poliner and Ellis 2005) approach. To develop a music recommendation system, Çataltepe and Altinel (2007) and Chordia et al. (2008) obtained melody information by pitch detection techniques, while Kuo and Shan (2002) mined the melody style by utilizing a chord assignment algorithm.

Rhythm refers to all of the temporal aspects of a musical work, no matter whether represented in a score, measured from a performance, or existing only in the perception of listeners (Gouyon and Dixon 2005). It represents a change pattern of musical events over time and conveys information such as tempo, meter, pitch duration, and harmonic duration (Tzanetakis and Cook 2002; Mitrović et al. 2010). An intuitive approach to describe the rhythm of musical data is to detect onset times and use the frequently occurring inter-onset intervals as cues (Gouyon and Dixon 2005; Casey et al. 2008; Celma 2010). A review

of automatic rhythm description approaches can be found in Gouyon and Dixon (2005). The music recommenders have been reported to extract rhythmic features via the log-scale modulation frequency coefficients (LMFC) (Zhu et al. 2006), beat histogram calculation (Çataltepe and Altinel 2007), or auto-correlation metrics (Donaldson 2007a).

Music Structure refers to the way how music materials are presented (Celma 2010). The structure in music arises from certain relationships between the elements, such as notes, chords, and so forth (Paulus et al. 2010). On the level of individual notes, structures can be characterized by timbre, pitch, and time intervals. Phrases, chords, and chord progressions form the structures on a higher hierarchical level. On the level of entire musical pieces the subdivision can be style-specific, such as the *A-B* themes exposition, development and recapitulation of a sonata form, or the *intro-verse-chorus-verse-chorus-outro* of pop music (Celma 2010). A general goal of structure analysis is to divide an audio recording into temporal segments with some internal similarity of consistency in music information (Casey et al. 2008). Paulus et al. (2010) gave an overview of state-of-the-art methods for computational music structure analysis. Bozzon et al. (2008) thought that exploiting similarities on semantic audio parts made the song similarity more meaningful to the user by providing a finer granularity analysis, such that they incorporated the semantic segmentation into their music recommendation system.

3.1.3 Cultural Metadata

Cultural metadata is textual data about musical content, implicitly produced by the environment or culture and usually gathered from the Internet, weblogs, and forums (Pachet 2005). Since the Web contains a large amount of music-related information, there has been much interest in the methods that automatically assign cultural metadata to music items (Casey et al. 2008). Most of the works depend on the textual information retrieval and filtering techniques, which have been widely used in the content-based recommendation systems for documents and websites (URLs) (Adomavicius and Tuzhilin 2005).

Whitman and Lawrence (2002) used music descriptions generated from community metadata to recommend similar artists. This approach sent the query to a web search engine, downloaded top pages, and extracted text and natural language features using a HTML parser. Then it created a vector representation for each artist with the Term Frequency Inverse Document Frequency (TFIDF) values (Baumann and Hummel 2003). Their

extracted data was time aware, reflecting changes in both the artist's style and the public's perception of the artist (Casey et al. 2008). Vembu and Baumann (2005) presented a recommendation system using Amazon web service interface to retrieve album reviews for artists.

3.1.4 Combining Metadata

Berenzweig et al. (2004) argued that the "ground truth" of music relationships was present in both the objective analysis and the cultural sensibility of music. A subjective evaluation in Bogdanov and Herrera (2011) revealed that combining acoustic metadata (e.g., timbre, tempo, and rhythm) and genre labels provided significant improvement in listeners' satisfaction compared with the approach based on single source of information.

There have been some works combining distinct sources of information for efficient content-based music recommendation. Baumann and Hummel (2005) exploited a multimodal similarity model (Baumann and Halloran 2004; Baumann et al. 2004) that linearly combined the audio-based track similarity with the web-based artist similarity. Pampalk and Goto (2007) adopted a weighted linear combination of audio-based and web-based artist similarity, where weights were adjusted by users. The audio-based artist similarity was aggregated from the similarities of tracks. According to Knees et al. (2006), one drawback of the linear combination is that it does not reduce the number of necessary computation between pairs of songs. As an alternative, Knees et al. proposed an approach to prohibit the similarity calculation for songs that were unlikely to be similar by taking into account additional artist information. More precisely, only the distance between tracks by similar artists were calculated. A similar approach was found in Pampalk and Goto (2006).

3.2 User Preference Modeling

Generally, a system can discern a user's preference via explicit or implicit inquiries. Explicitly, a user provides an example (e.g., a seed song or an artist) when the user has an initial idea of what he/she wants. Otherwise, the system models the user's preference implicitly from a list of items that the user liked before (K. Kim et al. 2008; Bogdanov et al. 2010). The former situation is straightforward for content analysis methods to determine user preferences. On the other hand, the list in the latter situation can be regarded as the accumulation of a user's preferences, and a representative value needs to be extracted.

Given a set of items to derive a user's preference, a simple approach is to represent the user as a single point in the feature space, such as the mean point of all items (Bogdanov et al. 2010). Bogdanov et al. (2010) also proposed to use the density of the user's preferences by employing a GMM. The advantage of the GMM approach is that the model took the relevance of the attributes within the user's preferences into account. A user may prefer many classes of music in terms of genres or instrumentation (Yoon et al. 2008). To analyze and represent the multiple preferences D.-M. Kim et al. (2007) proposed a dynamic k-means clustering algorithm where the number of clusters, k, was controlled by the range of clusters. If the range of some clusters exceeded the limit, k increased.

Usually, a user's music list is growing as the user downloads or listens to pieces of music. And the user's preference may change with time. Concerning these, recommendation systems need to pay attention to when a user listens to what music (Yoon et al. 2008). Yoon et al. (2008) extended the dynamic k-means clustering (D.-M. Kim et al. 2007) by introducing a decrease rate to analyze a user's preference change with time. In this approach, when a user had listened to new music, the importance of music which the user listened before would be decreased. The weighted average values of clusters (center of "mass") were used to represent the user's preference. Similarly in Chen and Chen (2001), each transaction was assigned a different weight, where the latest transaction had the highest one. Chang et al. (2011) expanded this method by taking extra count information into consideration. Their proposed method was termed as transaction-interest-count-interest (TICI), where the music group containing more accessed music objects in a transaction had a higher weight than other groups in the same transaction. Since TICI emphasized both the weight of time and the weight of count of music groups, it could decide the rank of the group weight more precisely.

A hierarchical user preference model can facilitate the process of matching music items against the user's preference. Hijikata et al. (2006) built a decision tree as the user profile based on the extracted features and allowed the user to edit it. This approach enabled the recommendation to prioritize on important feature parameters. For instance, one user who disliked fast tempo music, could use tempo as a first branching attribute to exclude undesired music. After being rejected from the candidate set, these songs were not considered for the next stage, which reduced the computation load on average.

3.3 Matching Items against User Preference

The content-based recommendation is to suggest the items that fall into the active user's taste. Given the item representation and the user profile, the process of matching music items against the user preference can be performed by either an information retrieval task or a classification task.

3.3.1 Information Retrieval Task

The recommendation problem can be treated as an information retrieval (IR) task, where the user's preference is treated as a query and the candidate items are scored with relevance/similarity to this query (Balabanović and Shoham 1997; Adomavicius and Tuzhilin 2005; Melville and Sindhwani 2010). After extracting a signature to represent each music item, this type of method computes the distance between the query and the candidates with a pre-designed similarity measure, sorts the similarity scores, and finally gives recommendations from an ordered list of items (Adomavicius and Tuzhilin 2005; Zhu et al. 2006). The similarity/distance computation approaches two cases: the vector representation and the model representation of items.

In the vector space, the similarity computation can be straightforward by means of distance functions, such as Pearson correlation and Euclidean distance (Chordia et al. 2008; Bogdanov et al. 2010; Dickerson and Ventura 2009). On the other hand, given the model representation, the similarity computation can use the Kullback-Leibler (KL) divergence (Bozzon et al. 2008), Earth Mover's Distance (EMD) (Logan and Salomon 2001) or Monte Carlo sampling (Aucouturier and Pachet 2002b). KL divergence is a non-symmetric measure of the difference between two probability distributions. EMD expands the KL divergence to the the comparison of mixtures of distributions (Magno and Sable 2008; Chordia et al. 2008). This measure is defined as the minimum amount of work needed to transfer one model into another (Logan 2002; Pampalk et al. 2005). Monte Carlo sampling treats cluster models as probability distributions from which samples are drawn. For each piece of music, a sample is drawn and the similarity is based on the log-likelihood that a sample is generated by the model (Aucouturier and Pachet 2002b; Pampalk et al. 2005).

In addition to the attempts to design a distance function, the literature has also reported other possibilities to facilitate similarity search. Some researchers used a Self-Organizing

Map (SOM) to create a lower dimensional space and simple distance metrics (Vembu and Baumann 2005; Dickerson and Ventura 2009). SOM is an unsupervised learning algorithm which creates a new n-dimensional space (usually two) from any higher dimensionality and preserves as much similarity among training data as possible. A SOM trained on song data (Dickerson and Ventura 2009) or artist data (Vembu and Baumann 2005) can be used to perform music recommendation according to the similarity. Cai et al. (2007) converted the recommendation problem to a scalable search problem, by building an efficient data structure via locality sensitive hashing (LSH). This music indexing approach organized music signatures based on hash codes generated by LSH, and returned the results sorted through a relevance-ranking function.

3.3.2 Classification Task

An alternative to IR is to treat the recommendation problem as a classification task, where a user's past opinions assign labels to the items (Adomavicius and Tuzhilin 2005; Melville and Sindhwani 2010). For instance, given a set of items labeled as "like" or "dislike" by the user, the system can classify other items into the two groups and only recommend the items in the group "like". Sometimes the classifier made use of a user's ratings by mapping the rating scale (from 1 to k) to k classes (Melville et al. 2002; Huang and Jenor 2004). In Huang and Jenor (2004), a user was asked to assign ratings ranging from one to five to the songs for training. Then the LBG vector quantization method was implemented to categorize a new song into a quantization group. As a result, the rating of the new song was assigned as the most common rating value in its group.

In the research area of music recommendation, there exist other proposals of multi-class classifier without the use of user rating values. Kuo and Shan (2002) used a Multitype Variant-Support (MTVS) classifier, and defined three types of recommendation (boolean recommendation, total rank recommendation, and total confidence recommendation) to deal with the multi-class problem. Based on the classification results, the recommendation engine calculated a ranking score for each candidate item to generate a ranked recommendation list. Çataltepe and Altinel (2007) used the Shannon entropy measure to determine the right clustering method (on which feature set to use). For each user, the clustering with minimum entropy was selected, for it was the best possible way to group the songs accessed by a user. After clustering, the recommendation system chose songs from each

cluster proportional to the number of songs that belonged to this cluster from the user history.

3.3.3 Hybrid of IR and Classification

Some work combined the above-mentioned two tasks to build a content-based music recommendation system. Zhu et al. (2006) proposed an integrated music recommender that included music genre classification, music emotion classification, and music similarity query. The classifiers first classified music into five genre classes and four emotion classes with AdaBoost algorithm, and then looked for similar songs from the class which the user's query belongs to. The hypothesis behind this approach was that "two similar songs must be in the same music genre class and the same music emotion". Based on the results of classification, their designed scheme could speed up the similarity query process without decreasing the precision.

3.4 Challenges

Content-based recommendation systems base their suggestion in the content analysis of items, ignoring the collaborative wisdom of crowds. They present several limitations that are described in the rest of this section.

3.4.1 Limited Content Analysis

Since the content-based techniques depend on the analysis of items, the quality of the available data is a determinant factor in the performance of such systems (Adomavicius and Tuzhilin 2005). With the editorial and cultural metadata the main problem is the inconsistency (Angeles et al. 2010). The effects of inaccurate metadata (e.g., spelling mistakes and subjective entries of genre) are numerous and there are no coordinated efforts to independently measure the (editorial/cultural) metadata quality (Freed 2006). For the acoustic metadata, only a relatively shallow analysis of certain kinds attributes can be supplied (Balabanović and Shoham 1997). Although there have been a large amount of research in extracting features from audio contents, it is still an open issue how to extract truly relevant and significant features (Celma 2010).

For the computational modeling of music similarity, numerous features can be taken into account. However, for a specific user at a specific time only a few features might be

sufficient. This is because people seem to choose music at one time from one perspective, such as lyrics, artists, instruments, or genre. And each of these descriptors may lie in a unique set of features. It is not clear which features are important and should be used without a comprehensive understanding of the user's demand. In my opinion, content-based music recommendation system can benefit from more accurate user preference models.

3.4.2 Scalability

Content-based approaches mainly use linear scan to look for proper recommendations, so that the processing time increases linearly with the data scale (Cai et al. 2007). To accelerate the computation on large-scale music collections, most audio-based approaches utilize track-level descriptions, such as one feature vector (Aucouturier and Pachet 2002a; Cano et al. 2005; Pauws et al. 2006) or one model representation (Knees et al. 2006) for a whole piece of music. Some approaches further group music pieces into clusters and perform searching on the cluster level (Huang and Jenor 2004; Logan 2004). However, such high-level descriptions lose the temporal characteristics and may not be able to provide enough information to characterize and distinguish various pieces of music (Cai et al. 2007). Besides, for the systems based on editorial/cultural metadata it becomes more difficult to maintain the consistency when catalogues become very large (Freed 2006).

3.4.3 Overspecialization

Since the goal of most content-based recommendation systems is to find music similar to a user's preference, finding novel and diverse music becomes an unavoidable challenge (Balabanović and Shoham 1997; Cai et al. 2007). When a system can only recommend items that are similar to those already accessed, the novelty and diversity of recommendation will be low (Adomavicius and Tuzhilin 2005; Pampalk et al. 2005). For example, a listener with no experience with Soul music will never receive a recommendation for even the greatest Soul music. This problem can be addressed by introducing some randomness (Balabanović and Shoham 1997) or using other factors to promote the eclecticness of the recommended items (Celma 2010). In the context of information filtering, the randomness can be introduced by the crossover and mutation operations (as part of a generic algorithm) (Sheth and Maes 1993). To reach a compromise of similarity and novelty, Çataltepe and Altinel

(2007) recommended a certain percentage of songs according to the user's preference, and the remaining songs from a repository of the popular songs at the time.

3.4.4 New User Problem

Since the content-based filtering is achieved by analyzing items' contents rather than exploring other users' opinions, new items with few ratings can avoid being excluded from discovery. However, the cold-start problem may occur when a new user enters the system (Adomavicius and Tuzhilin 2005; Celma 2010). Unless the new user explicitly clarifies a demand, the system cannot adapt to the user preference with few listening histories such that the user will not be able to receive reliable recommendations (Balabanović and Shoham 1997).

3.4.5 Gray Sheep

The problem of *gray sheep* (users with atypical tastes) can occur too, if the size of collection is not big enough or the collection is biased towards the mainstream music (Celma 2010).

Chapter 4

Extending the Capabilities of Traditional Recommenders

Traditional music recommendation systems depend on collaborative filtering or content-based filtering to generate recommendations. So far, the research community has broadened its attention to include other aspects, as listed in Table 4.1. Hybrid approaches combine the collaborative filtering and content-based filtering together to leverage the strengths and weaknesses of each approach. User modeling aims to develop a better user profile. Context awareness associates users and items in a specific circumstance, such as working or dancing. Tag-based recommendation labels items with users' opinions. Recommendation in the Long Tail tries to minimize the popularity bias and mostly accompanies collaborative filtering, for content-based filtering ignores item popularity at all. Recommendation networks introduces some new properties to the recommendation strategies. Visualized recommendation provides visual perception to users and sometimes allows interaction. Playlist generation can be deemed as a variation of top-N recommendation, satisfying the needs specified by users. Group recommendation involves some pre- or post-processing by either aggregating multiple user preferences into a unit user profile or uniting separate recommendation results into one recommendation list.

Out of all the works in the bibliography, 116 works in total tried to extend the capabilities of music recommendation systems. Most of these works can be identified with one extension approach, while 11 works can be tagged with two labels. For example, the work by van Gulik and Vignoli (2005) covered both the visualized recommendation and the playlist generation. Fig. 4.1 shows the approximate categorization of the 116 extension

works. Playlist generation and context-aware recommendation respectively accounted for around one-quarter of all works. On the other hand, only 2% or research works proposed to recommend niche items in the Long Tail. The other approaches took up 6% to 13% of works. So we can conclude that more than half of the research effort has been for the playlist generation and the context-aware recommendation. Fig. 4.2 depicts a detailed model of music recommendation systems. The following sections will discuss the details of each extension approach.

Table 4.1: Extensions to traditional recommenders

Extensions	Description
Hybrid collaborative and content-based	Combine collaborative filtering and content-based filtering into a hybrid recommendation strategy.
User modeling	Customize user profile with music-related information.
Context-aware	Take into account additional contextual information (e.g., time and location) when providing recommendations.
Tag-based	Apply tags to describe items (e.g., artists and songs).
Recommendation in the Long Tail	Recommend unpopular items to minimize popularity bias in collaborative filtering.
Recommendation Networks	Explore music by navigation under links between items and/or users.
Visualization	Provide an interface visualizing a music collection or recommendation results.
Playlist	Recommend an ordered (or unordered) list of music.
Group recommendation	Make recommendations to a whole group of users.

4.1 Hybrid of Collaborative and Content-based Filtering

Both collaborative filtering and content-based filtering have considerable merits and drawbacks. Collaborative filtering can reflect a cultural knowledge about music and capture the dynamics of music tastes as the population changes its sensibilities (Donaldson 2007a). It works well in domains where it is hard to analyze content information by automated

Fig. 4.1: Extensions of music recommendation systems.

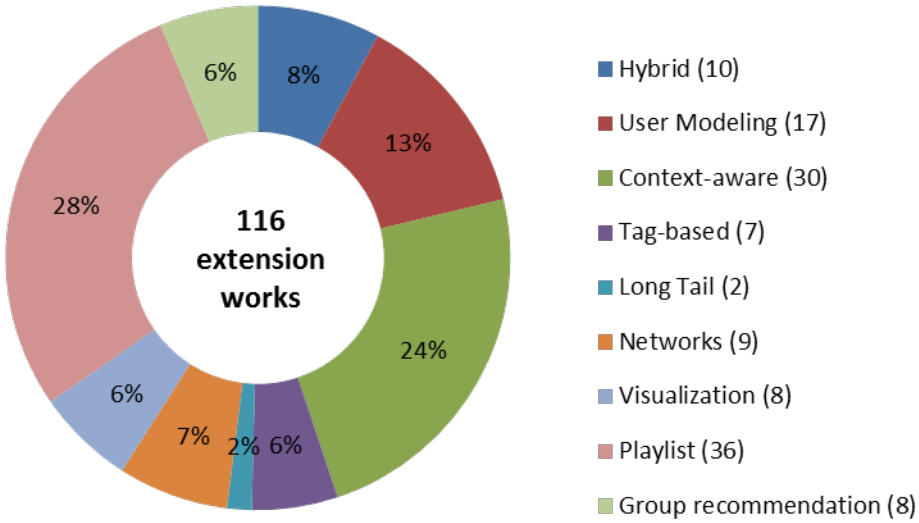
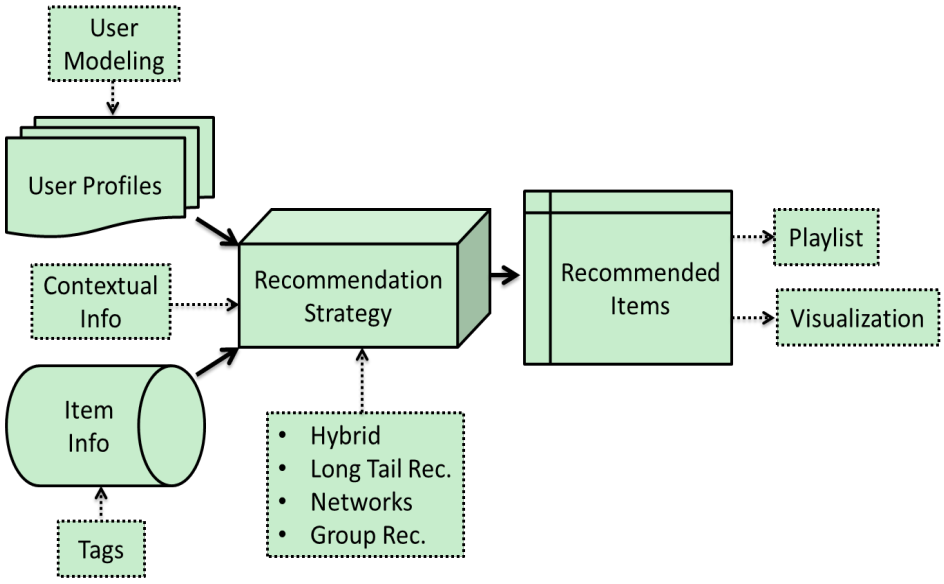


Fig. 4.2: A detailed model of music recommendation systems.



processes (Melville et al. 2002), and has the ability to provide serendipitous recommendation. But this method is essentially content-blind, such that it has the new item problem, popularity bias, and so forth (Aucouturier and Pachet 2002a). By contrast, content-based

filtering does not necessarily need user feedback to characterize a given song. Since recommendations of music are based on content directly, it can recommend completely new items and solve the popularity bias (Celma 2010). Unlike collaborative filtering where the popularity information of music is constantly changing over time, the content or metadata of music (e.g., artist, melody, and rhythm) do not change over time. Therefore, one musical item can be analyzed only once. However, content-based approaches cannot discriminate popular music from unpopular music, nor can it detect any cultural information (Donaldson 2007a). The two techniques have complementary characteristics as listed in Table 4.2.

Table 4.2: A comparison of collaborative filtering and content-based filtering

	Collaborative filtering	Content-based filtering
Social behavior analysis	Yes	No
Content analysis	No	Yes
Diversity & Serendipity	Yes	No
Overspecialization	No	Yes
New item problem	Yes	No
Popularity bias	Yes	No

In order to leverage the strengths and weaknesses of collaborative filtering and content-based filtering, there have been some recommendation systems combing the two methods into a hybrid approach (Adomavicius and Tuzhilin 2005; Melville and Sindhwani 2010). Burke (2002) gave a survey of hybrid recommender systems and categorized the methods to integrate different approaches into a hybrid recommender, such as weighted, switching, mixed, cascade, and so on. I will now describe the hybrid approaches that have been utilized by music recommendation systems.

4.1.1 Weighted Approaches

One simple way to build a hybrid recommendation system is to allow collaborative filtering and content-based filtering produce recommendations separately and then merge them together using methods such as linear combinations (Burke 2002; Celma 2010). Tiemann et al. (2007) investigated ensemble learning methods for hybrid music recommendations. In this approach, collaborative filtering and content-based filtering each produced a weak learner, whose results were combined by a simple combination rule. Lu and Tseng (2009) presented

a music recommendation system that weighted collaborative filtering, content-based filtering, and emotion-based recommendation. For each user, the system first computed some initial weights of each recommendation method through a user survey. Then in the process of making recommendations, the system collected the user's feedback and adjusted the weights of the three recommendations to adapt to the user's variations of interests.

4.1.2 Feature Augmentation

To overcome the problems of collaborative filtering, such as sparsity, cold start, and popularity bias, some hybrid systems are based on traditional collaborative filtering but also maintain a content-based profile for users or items. These content-based profiles are then used to calculate the similarities between users or items to facilitate recommendation. Cohen and Fan (2000) described a collaborative filtering spider that collected lists of semantically related entities from Internet. They extracted information about genre and musical style of each artist from *Allmusic*[1] and created a pseudo-user for each genre. The pseudo-user rated everything in the genre positively and everything not in the genre negatively. Then the system used the standard k-nearest neighbor approach to find the genres that correlate with the user's preferences and recommend artists in these genres. Their results suggested that collaborative filtering methods may be useful even in cases where there was no user data at all.

On the other hand, user access patterns discovered by item-based collaborative filtering are also useful to determine music similarities, such that they can be incorporated into a content-based recommendation system (Shao et al. 2009). This approach is essentially a dynamic music similarity measurement scheme. The user access patterns assumed that two pieces of music with a higher co-occurrence frequency were more similar to each other in human perception. Shao et al. utilized the user access patterns to dynamically learn weights for each content features and then performed music recommendation based on music similarities. It has been argued by the authors that "comparing with other probabilistic models and hybrid approaches, our method incorporates the content similarity data and collaborative filtering information seamlessly."

[1] http://www.allmusic.com/

4.1.3 Mixed Approaches

Some recommenders do not only combine, but expand the description of the data sets by taking into account the ratings of users and the description of the items, which is called the mixed approach (Burke 2002; Celma 2010). Li et al. (2004, 2005, 2007) transferred audio features into aggregate features using a K-means clustering algorithm and attached those aggregate features to the user-rating matrix. Based on this extended rating matrix, item community was derived from the Pearson correlation and the k-Medoids clustering algorithm. Donaldson (2007a) merged the co-occurrence information of music and the acoustic features into a unified scale. The co-occurrence of music in playlists was presented in an adjacent matrix, where every column or row represented a song and each cell contained the number of co-occurrence times. From this matrix, spectral graph eigen-vectors were extracted and merged with the acoustic feature vectors to identify k-nearest neighbors. This ability enabled the system to disambiguate the music seeking behavior of a given user if they supply a given playlist as a query.

Some works attempt to directly develop a unifying model to combine collaborative filtering and content-based filtering. Popescul et al. (2001) extended Hofmann's aspect model (Hofmann 1999) to incorporate three-way co-occurrence data among users, items, and item content. This model introduced a set of latent variables to explain the generative process for the observed data (Celma 2010). Based on it, Yoshii et al. (2006, 2008) presented a probabilistic latent semantic indexing (pLSI) for hybrid music recommendation that incorporated both rating data provided by users and content-based data extracted from audio signals. Yoshii et al. (2007) improved the efficiency and scalability of the previous approach using incremental learning. The original pLSI treated users and items as discrete random variables that follow multinomial distributions, while Yoshii and Goto (2009) extended this work to a continuous pLSI that incorporated GMMs.

4.2 User Modeling

The main goal of a music recommendation system is to suggest music that the listener will be interested in. Understanding users can be crucial to proposing a more reasonable recommendation than applying the same recommendation strategy to all users. For example, some users show a clear bias towards style when choosing music, and some emphasize timbral similarity, while others are interested in finding new and unexpected music (Bau-

mann et al. 2004). Moreover, users' feelings and expressions can be different towards the same music (Kodama et al. 2005). Most recommendation systems try to understand a user based on two complementary types of information: psychological factors (personality, demographics, social relationships) and explicit musical preferences (Perik et al. 2004; Celma and Serra 2008). Those psychological factors are rarely updated or modified, while the explicit musical preferences are updated more often (Celma and Serra 2008). These two aspects are not isolated but correlated. As pointed out in Uitdenbogerd and van Schnydel (2002) and Lesaffre et al. (2006), the music preference of a user is dependent on the information of age, origin, occupation, socio-economic background, personality factors, gender, and so on. The incorporation of demographic data into recommenders has proved to be useful in improving the performance of music recommendation systems (Yapriady and Uitdenbogerd 2005). Customizing the user profile is a field of increasing importance for information filtering, which can provide adaptive recommendation service for users with different backgrounds, experiences, and tasks performed (Vassileva 1994). Using user models in recommendation systems can reduce the search space and make recommendation services more accurate and easier to use (Chai and Vercoe 2000).

In the music recommendation field, there have been a few attempts to set up user profiles with music-related information (Celma 2010), including User Modeling for Information Retrieval Language (UMIRL) (Chai and Vercoe 2000), MPEG-7 (Tsinaraki and Christodoulakis 2005), Friend of a Friend (FOAF) (Celma et al. 2005; Celma and Serra 2008). UMIRL is an XML-like language, specially designed for music information retrieval systems. It describes a user with indirect/demographic information and direct information relating to music interests (e.g., favorite artists or genres). It even allows a user to add her own definition to music, for example, "romantic music for me means slow music with titles or lyric including word love." The MPEG-7 is an ISO/IEC standard which provides a set of description schemes to describe multimedia assets. For the user preferences of multimedia content, it includes the user's history of interacting with multimedia items, searching preferences (e.g., country, language, and artist), and keywords (e.g., dramatic and fiery). FOAF, which is based on RDF/XML vocabulary, may include demographic (e.g., name, gender, and age), geographic (city, country, latitude, and longitude), and social (relationship with others) information together with usage patterns and music preferences. Other approaches to model user preference also exist. *Scrobbler*[2] builds a user profile from the

[2]http://www.last.fm/about

usage records. Each time a song is played, a little note containing the artist's name, the song title, and a timestamp, will be sent to *Last.fm*. In this way, the user profile reveals "what songs you play most often, which songs you like the most, how much you've played an artist over a certain amount of time, which of your friends have similar tastes... all kinds of stuff". Using a list of songs which a user listened to in the past, content analysis techniques can further grasp the user's preferences with respect to music attributes (Hoashi et al. 2003; Hijikata et al. 2006; K. Kim et al. 2008).

Usually, a user's preference shifts with time, in terms of years, seasons, days, and even hours (Hayes 2003; Yoon et al. 2008; Park and Kahng 2010; Hu and Ogihara 2011). The recommendation system needs to focus on when the user listens to what music. For example, a user who liked calm and soft music before, may like noisy music now. One strategy considering the temporal effect on music preference is to decrease the importance of former data (Yoon et al. 2008). However, the consumption behavior of music is different from that of others like movie, books, and games, so that unlike these other contents, people listen to songs repeatedly and continuously (Celma and Lamere 2011; Celma 2010; Park and Kahng 2010). Park and Kahng (2010) studied the periodicity of music listening along the time axis, such as weekly periodic patterns and daily cycles, and their analysis of the seasonal effect showed that some kinds of music were preferred at particular seasons or months. For example, dance music is more popular in summer than winter. Although the number of times a user plays a song shows his/her preference for that song (Park and Kahng 2010; Hu and Ogihara 2011), the preference and the playcounts are not linearly correlated (Park and Kahng 2010).

Since the modeling of user preference is not simple or straightforward, user feedback and interaction has become a powerful auxiliary tool (Hoashi et al. 2003; Vignoli and Pauws 2005; Hijikata et al. 2006; Pampalk and Gasser 2006; Celma and Cano 2008). Alternatively, tracking a user's participation in a playing song can be used to evaluate the user's attitude towards that song (Hu and Ogihara 2011). If a user listens to a song from beginning to end, the user is likely to like that song. On the other hand, if a user skipped a song before playing 5% of the length, the user is assumed to dislike the song (Hu and Ogihara 2011). Collecting these data can inform the system whether the current recommendation is effective or not (Pampalk et al. 2005). Pampalk et al. started the playlist generation with an arbitrary song automatically selected and then adjusted the playlist according to

users' skipping behavior. This scheme can help to solve the problem of cold start when a recommendation system has no reference as to what kinds of songs users like or dislike.

4.3 Context-aware Recommendation

The traditional recommendation approaches, either collaborative filtering or content-based filtering, operate in a two dimensional $User \times Item$ space, and do not take into account any additional contextual information, such as time, location, and activity (Adomavicius and Tuzhilin 2011). In other words, they do not put users and items into context when providing recommendations. Dey (2001) defined the context as "any information that can be used to characterize the situation of an entity such as a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves". According to some studies of the psychology and sociology of music (North and Hargreaves 1996; North et al. 2004; McEnnis and Cunningham 2007), a user's short-term music preferences are associated with the user's context, such as emotions, activities, or external environment. For example, the music for a dinner can be different from the music for a dance party on musical attributes such as tempo and rhythm. Even for the same activity, for example working, a user's music preference on a rainy Monday morning may be different from that on a sunny Friday afternoon. Moreover, music preferences are affected by the season (Lee et al. 2010; Park and Kahng 2010), such that a user may listen to different music in summer and winter. Those traditional recommendation systems that ignore contextual information may not be able to accurately capture a user's temporary preference.

Context-awareness is to use the information about the circumstances to provide relevant information and/or services to the user (Cuddy et al. 2005). The usage of contextual information in recommender systems can be traced to the work by Herlocker and Konstan (2001), who hypothesized that the inclusion of knowledge about the user's task into consideration can improve recommendation performance. Researchers in music recommendation have recently paid more and more attention to the context-aware music recommendation in order to provide more satisfying music to users. For example, *Sourcetone* radio[3] takes the specified mood of a listener into the consideration when streaming music. *Songza*[4]

[3]http://www.sourcetone.com/
[4]http://songza.com/

recommends various playlists based on the time of day and activities (e.g., working or cooking). The rest of this section will focus on how to obtain the contextual information and incorporate them into a music recommender system.

4.3.1 Obtaining Contextual Information

Existing context-aware music recommenders have explored many kinds of context information, such as location (Park et al. 2007; Lehtiniemi 2008; Lee and Lee 2007; Rho et al. 2009; Kaminskas and Ricci 2011), time (Park et al. 2007; Lehtiniemi 2008; Liu et al. 2009; Su et al. 2010), weather (Park et al. 2007; Su et al. 2010), emotions (Kodama et al. 2005; Kuo et al. 2005; Park et al. 2007; Dornbush et al. 2007; Rho et al. 2009; Shan et al. 2009; Han et al. 2010), physiological states (Wijnalda et al. 2005; Oliver and Kreger-Stickles 2006; Niitsuma et al. 2008; Liu et al. 2009; Su et al. 2010), and tasks/activities (Cai et al. 2007; Rho et al. 2009). These contextual information can be obtained via either explicit or implicit methods (Adomavicius and Tuzhilin 2011).

Explicit approaches gather contextual information by directly approaching users (Adomavicius and Tuzhilin 2011). For example, a music website may obtain contextual information by asking users to answer some specific questions before proving access to music. Baltrunas et al. (2010) asked the users to indicate the appropriate context for their rated item, in addition to a standard rating matrix. The contextual dimensions included activity (work, party, relaxation, and sport), weather (sunny, rainy, cold), time of day (morning, noon, evening), the valence mood (happy, sad), and the arousal mood (calm, energetic). Based on these information, they predicted the best context for each item.

Implicit approaches gather contextual information from the data or the environment without interaction with the user. For example, J.-H. Kim et al. (2006, 2007, 2008) collected user information (e.g., sex, age, and pulsation) and surrounding contexts (e.g., weather, temperature, and location) from automatic sensing data based on an open service gateway initiative (OSGi) framework. They traced user sex, age, and location information using an RFID Tag which was attached to a user's watch, and obtained temperature information from a temperature sensor through real-time Zigbee communication. Lee and Lee (2006, 2007) obtained the weather data from the Weather Bureau, which contained the data such as season, month, a day of the week, atmospheric conditions, and lowest/highest/average temperatures.

Sometimes, the user's raw context data cannot be applied to recommendation systems directly, but needs to be abstracted into a concept level. For example, the raw context data, 14:00, 37°C, may indicate a hot afternoon in summer. But the context abstraction can be fuzzy when the contextual concepts do not have the clear boundary. For example, there is no clear boundary between spring and summer in terms of a day, and a temperature value of 27°C can be either warm or hot. To solve this problem, Park et al. (2007), Shin et al. (2009), and Lee et al. (2010) utilized the fuzzy set theory with each contextual concept considerered as a fuzzy set. Guan et al. (2006) employed the native Bayesian algorithm to automatically derive the mood of users from the contextual information. In order to automatically suggest music when users read Web documents, Cai et al. (2007) proposed an emotional allocation model to abstract emotions from textual information of Web documents and music metadata.

4.3.2 Incorporating Context in Recommendation Systems

Incorporation of context converts a two-dimensional (user and item) problem into an n-dimensional one through the addition of several contextual dimensions or variables (Adomavicius and Tuzhilin 2005). Here, I will review some approaches that exist in the literature of music recommendation. For a more comprehensive discussion of how to model context in recommender systems, please refer to Adomavicius and Tuzhilin (2011).

Some systems treat the context-aware recommendation problem as the recommendation via context-driven querying and search (Adomavicius and Tuzhilin 2011). The systems of this type, such as *Songza*, typically use contextual information (e.g., current mood or location) to query or search a certain repository of music and present the matching items to the user. J.-H. Kim et al. (2006, 2007) prepared music profiles using contextual labels such as weather, temperature, location and user age, and stored the labeled profiles in a music content information database. Then, a content-based statistical filtering method would select suitable music that matched the user's need. Shin et al. (2009) introduced an intermedium layer of context concepts to compute the similarities between a user's current context and an item. The system multiplied the correlation between a user's current context and the context concepts with the correlation between the context concepts and an item.

Some researchers employed the reasoning methodology to achieve context-awareness. Lee and Lee (2006, 2007) used the case-based reasoning (CBR) on the user's demographics, behavioral patterns, and context. The fundamental concept in CBR is that similar problems

will have similar solutions, such that the system solves a new problem by analyzing the solutions to previous, similar problems. For example, a user who usually listened to the song "Who'll Stop the Rain" by Creedence Clearwater Revival on a rainy day, will likely to listen to this song on a rainy day in the future. For a specific user in a specific context, their system recommended the music that the similar users listened most in the similar contexts. As an alternative, Rho et al. (2009) used rule-based reasoning to reason about the context information and the user's favorite mood from the user profile information. Kuo et al. (2005) and Shan et al. (2009) proposed the music affinity graph algorithm to discover the relationship between music features and emotions from film music.

Sometimes, the information about the current context can be used for selecting or constructing the relevant set of data records (Adomavicius and Tuzhilin 2011). Chedrawy and Abidi (2006) presented a context-based collaborative filtering algorithm, where the similarity between items was computed over the user-selected perspectives only, such as lyrics, tunes, vocals, etc. Alternatively, Su et al. (2010) grouped users under similar context conditions to improve collaborative filtering. Cebrián et al. (2010) generated a segmentation in context space of the user profile into several contextual profiles. When a user requested a recommendation, the system used the context-constrained micro-profiles that matched the current temporal context to produce a recommendation list via a collaborative filtering approach. Lee et al. (2010) described four ways to capture context-dependent preferences of users in the framework of collaborative filtering.

The literature refers to works that directly used the contextual information in the modeling techniques. Dornbush et al. (2007) modeled the complex relationships between a user's musical preferences and physiological states via various machine learning algorithms, such as decision trees, AdaBoost, support vector machine (SVM), k-nearest neighbors, and neural networks. The system was trained to understand what music would be preferred under what conditions. J.-H. Kim et al. (2008) performed a modeling for each item using a hidden Markov model (HMM) based on the context information. This context-based HMM expressed user history according to the selected items and assisted in the recommendation service together with collaborative filtering. Hu and Ogihara (2011) used GMM to represent the time pattern of listening and compute the probability of playing a song at that time.

4.4 Tag-based Recommendation

Tags are free text labels that are applied to items such as artists, albums, and songs. Unlike traditional keyword assignment, no restrictions are placed on the makeup of a tag. Additionally, there is usually no restriction on the number of tags that can be assigned to an item. These tags are generally assigned by a non-expert easily. Social tags are the aggregation of individual sets of tags, which can provide information about various features, like genre, style, mood, users' opinions, or instrumentation (Lamere 2008). Social tagging is the process to annotate items with free text labels (Symeonidis et al. 2008; Nanopoulos et al. 2010). It addresses the limitations of centralized metadata by providing the opportunity of describing content to public communities of users (Casey et al. 2008). Social tagging is a key part of Web 2.0 technologies and shows its power when the tags of many users are aggregated and shared (Eck et al. 2007; Symeonidis et al. 2008; Nanopoulos et al. 2010). The increasing number of users tagging their accessed items has facilitated the emergence of tag-based profiling approaches, which has become an important source of information for recommendation (Milicevic et al. 2010).

In the domain of music, Web sites such as *Last.fm*[5] provide the possibility that users can assign as many tags as they want to any track, album, or artist in their profiles. The social tags not only carry useful information about the musical items, but also about the users who provide them. It is assumed that the tags assigned to items usually include the information related to their contents. Analogously, the tags provided by users can describe their interests, tastes and needs (Cantador et al. 2010). With social tags, users can find items more easily (Firan et al. 2007; Levy and Sandler 2009), and systems can improve the search mechanisms and the personalized music recommendation (Symeonidis et al. 2008; Green et al. 2009; Milicevic et al. 2010).

4.4.1 Incorporating Social Tags in Recommendation Systems

A simple way to utilize tags in recommendation is to consider them as query terms, e.g., find all songs tagged as "pop". The tag-based searching approach has the significant advantage of computation speed and is able to return results instantly (Firan et al. 2007). In the literature, there are also works that incorporated the tag-based user/item profile into collaborative filtering (Firan et al. 2007) and content-based filtering (Cantador et al.

[5] http://www.last.fm/

2010). Firan et al. (2007) defines user profiles as collections of tags together with corresponding scores representing the user's interest in each of these tags. In their approach, the collaborative filtering worked on a User-Tag matrix and generated a list of recommended tags based on what tags other similar users have used. With this list of tags, the top 10 tracks that have been tagged with most of these tags were returned as recommendations. Cantador et al. (2010) investigated the issue of applying tag-based profiles to both users and items, and studied several weighting schemes to measure the importance of a given tag for each user and item. The tag-based user/item profiles were then exploited by content-based recommendation models. McFee and Lanckriet (2009) used tag-based artist profiles together with artist-level acoustic features and biography summaries to reproduce human-derived measurements of subjective similarity between artists.

Social tags can reveal the three-dimensional correlations between user-tag-item, which project a user's perception of a particular musical item (Symeonidis et al. 2008; Nanopoulos et al. 2010). To capture the multimodal perception of music by particular users, Symeonidis et al. (2008) developed a unified framework to model the social tagging data with three-order tensors, and performed the latent semantic analysis (LSA) on this model to discover the latent structure using higher-order single-value decomposition. Recommendations were generated according to the user's multimodal perception of music based on the discovery of the latent structure. To further improve the quality of recommendation by addressing the problem of data sparsity, Nanopoulos et al. (2010) exploited similarities between the music items that were computed based on audio features.

4.4.2 Challenges

Even though social tagging has provided some new possibilities to perform the task of recommendation, the tag-based systems still need to deal with some challenges posed by the nature of tags. First, the free nature of social tags poses the vocabulary problem, such as polysemy and synonymy (Lamere 2008; Nanopoulos et al. 2010). Polysemy refers to the problem that tags have more than one meaning. For example, the tag "classic" may be assigned to both the music composed in 18th century and the rock songs from the 60's. Whilst synonymy refers to the problem that different tags have the same or similar meanings, e.g., "rnb", "r and b", and "r&b". To address the vocabulary problem, Levy and Sandler (2008) applied LSA to social tags.

Second, social tags may have the problem of data sparsity because some items may be tagged poorly. New items may not be retrieved because they do not have enough assigned tags. These problems can be addressed by employing other ways to collect tags, such as deploying annotation games or autotagging audio content (Turnbull et al. 2008). Tagging games like *Tag a Tune*[6] collect tags with human players, while the autotagging approaches use automated content analysis to apply tags. The general idea of autotagging is to map the content-based features to a more semantically meaningful space, which can be formulated as a multi-label classification problem (Ness et al. 2009). In Sandvold et al. (2006), Eck et al. (2007) and Ness et al. (2009), tags were automatically predicted from audio features with supervised learning. Moreover, Ness et al. (2009) used stacked generalization to exploit possible correlations between tags. The output autotags can furnish the information of musical items that lack descriptive tags and assist recommendation.

4.5 Recommendation in the Long Tail

The statistical analysis of music transactions/plays has revealed a popularity decreasing curve across titles, which is often termed as Long Tail (Anderson 2004; Elberse 2007; Celma 2010). Except the head where a small number of popular items (the hits) are located, the Long Tail is relatively flat in the shape of the frequency distribution. It illustrates that a large number of songs or artists, the misses, are only played or downloaded by a relatively small group of people. It seems that a majority of users prefer popular titles, while only a minority exploit niche titles. The characteristics of the Long Tail is shaped by the popularity of items (Celma and Cano 2008). To measure the popularity bias, the Gini coefficient, which is usually used to measure the inequality of a given distribution, has been applied (Elberse 2007; Fleder and Hosanagar 2007). When every item is equally popular, the Gini coefficient equals zero. On the other hand, the Gini coefficient reaches one when the distribution concentrates on one item. In Elberse (2007), the Gini coefficient of *Rhapsody* music data is as high as 0.838. In some sense, this value infers that the music industry is dominated by a high popularity bias. The 2007 "State of the Industry" report (Soundscan 2007) showed that among the 844 million digital tracks sold in 2007, only 1% of all digital tracks accounted for 80% of all track sales. Also, 1,000 albums accounted for

[6] http://www.gwap.com/gwap/gamesPreview/tagatune/

50% of all album sales, and 450,344 of the 570,000 albums sold were purchased less than 100 times.

In the old days of brick-and-mortar stores, the hit-driven economics of the industry endorsed that a small inventory of popular items would satisfy most customers (Anderson 2004; Goel et al. 2010). There is not enough shelf space for all of the CDs and not enough radio waves to play all the music that is released. But it is believed by some economists that the Long Tail can be profitable in e-commerce by "selling less of more" (Anderson 2004; Brynjolfsson et al. 2006). In fact a sizable fraction of total consumption is generated from niche items not available in traditional brick-and-mortar stores (Anderson 2004; Goel et al. 2010). Studies have shown that nearly everybody's taste deviates from the mainstream somewhere, with most people consuming niche products at least some of the time (Anderson 2004; Elberse 2007; Goel et al. 2010). Interestingly, an analysis of *Yahoo! Music* data showed that "both the most popular and the least popular songs receive the highest ratings, with a dip in the middle of the inventory", and that "the most obscure songs receive slightly higher average ratings than the most popular ones" (Goel et al. 2010). This contradicts the "Double Jeopardy" (obscure items are not generally known, and they are not generally liked by those who know of them) supported by the analysis of DVD rental data (Elberse 2007; Goel et al. 2010). In movie datasets, average consumer rating values increase with popularity. In any case, aggregating the Long Tail has led to business successes, such as *Google*, *eBay*, *Amazon*, *Netflix*, and *Rhapsody* (Anderson 2004). Moreover, according to Goel et al. (2010) the tail availability may even boost the head sales. The essence of the Long Tail business is to offer customization to individual consumers (Elberse 2007). In the music industry, although the hits will continue to dominate consumption (Elberse 2007), driving demand down the Long Tail can potentially create a larger market overall (Anderson 2004).

With the development of advanced tools for search and recommendation, listeners have a better chance to discover and enjoy a wider range of music that may be less popular but a good match to personal taste (Levy and Bosteels 2010). A study on *Last.fm* data showed that the item popularity tended to reinforce popular music at the expense of discarding less-known music (Celma and Cano 2008). The navigation guided by popularity may be stuck in the head, such that it is not easy to reach a niche item from a hit (Celma and Lamere 2011). To improve the visibility of Long Tail artists, Levy and Bosteels (2010) built a prototype recommender for long tail artists using conventional item-based collaborative

filtering. However, it is notable that the popularity effect can drastically filter out all low-quality items in the Long Tail, since poor-quality tracks are hardly popular (Celma and Cano 2008). In some sense, the popularity effect guarantees the data quality of the recommended tracks. The user-centric experiment with 288 subjects and 5,573 tracks (Celma and Herrera 2008) indicated that collaborative filtering outperformed content-based methods in terms of users' perceived quality even though it recommended less novel items (on the Long Tail). Therefore, to assist in this discovery task, the recommendation techniques are needed to discount popularity on an appropriate level (Fleder and Hosanagar 2007).

4.6 Recommendation Networks

Given a database of music, a music recommendation system is constructed to assist users to explore music by navigation under guided links among items or users (i.e., item-to-item, user-to-user, or user-to-item) (Cano et al. 2005; Seyerlehner et al. 2009). These guided links constitute the information that music recommendation engines use to produce results (Buldú et al. 2007). Recently there have been some works that made use of complex networks (Anglade et al. 2007a; Buldú et al. 2007; Donaldson 2007b; Fields et al. 2008) and music similarity networks (McFee and Lanckriet 2009; Seyerlehner et al. 2009) to assist the music recommendation. The motivation behind this attempt is that the properties of musical networks contain the information that can be used for the development of a music recommendation system (Buldú et al. 2007).

Complex network theory is derived from graph theories (Albert and Barabási 2002). It was realized from the experiment of Milgram (1967). He sent several packages to randomly chosen people, asking them to forward it to a given person. If those people did not directly know the recipient, they should "mail this folder to a personal acquaintance who is more likely than you to know the target person" (Milgram 1967). In Milgram's experiment, most of the letters never arrived at their destinations, but of the ones that did, it took an average of six forward to get there. This is the origin of the six degrees of separation theory: assuming that people are all connected through chains of acquaintances, the theory suggests that two people on the planet are, at most, six handshakes away from each other. So far, complex networks have been applied to many problems including music topology analysis (Gleiser and Danon 2003; De Lima e Silva et al. 2004; Lambiotte and Ausloos 2006; Park et al. 2007).

A complex network is a group of nodes (vertices) connected via links (edges). Directed links connect directed networks while undirected links connect undirected networks. If a node has more links to other nodes, it has a higher probability to connect a new node, and it is more likely to become central for the network. A weighted network is connected by links that carry a numerical value measuring the strength of the connection. A detailed description of complex network can be found in Albert and Barabási (2002). For the recommendation systems, there are two distinct kinds of nodes, namely users and items. These two groups of nodes can lie in two parallel planes (called bipartite graphs), connected to each other with certain links. This bipartite structure can be projected to either user plane or item plane. Zanin et al. (2008) described how to build a recommendation system from the complex networks.

With the use of complex analysis, researchers have uncovered some clues to the structure of recommendations. Cano et al. (2005) analyzed the networks of artists where a link was assigned if two performed or composed similar music. The information was extracted from online music portals such as *AllMusic*[7] and *Yahoo! Music*[8]. They found small-world effect in all networks, which had an influence on the navigation through the network. By taking into account all uploaded playlists, Buldú et al. (2007) and Donaldson (2007b) created a network of songs according to the information of co-occurrence. When two songs co-occurred in playlists, an undirected link was created between them, and the connection weights were assigned according to the number of times that the two songs appeared together. Such networks offered the association data between songs. Furthermore, Buldú et al. (2007) studied the evolution of a network with time and characterized the its growth under musical tastes. Anglade et al. (2007a) created virtual communities of users in peer-to-peer systems using a complex network theoretic approach.

For the music similarity networks, Seyerlehner et al. (2009) argued that any recommender system could be transformed into a recommendation network or graph under some restrictions, such as independence of user profiles. Thus, they constructed a recommendation/browsing graph to analyze the performance of a content-based music recommender system and assist the modification of the system design. Fields et al. (2008) tried to improve navigation through music collections using both social metadata and content-based similarities. They combined the complex network theory, network flow analysis and signal-

[7]http://www.allmusic.com/
[8]http://music.yahoo.com/

based music analysis to construct an artist similarity network that explored the relationship between the connectivity of pairs of artists on the *Myspace*[9] top friends network and a measure of acoustic dissimilarity of these artists. This work was employed in playlist generation (Fields 2011). Considering the variety and complexity of relationships among objects in music social communities, Bu et al. (2010) and Tan et al. (2011) modeled multiple kinds of social media information and acoustic features into a hypergraph. The hypergraph was used to model high-order relations, where a hyperedge represented a set of objects. On this hypergraph, recommendation was considered as a ranking problem.

4.7 Visualized Recommendation

The visualization for a music collection can provide an interface with comprehensive and functional interaction to users. Automatic music analysis and music similarity measurement have brought various interfaces with novel visualization functions (Casey et al. 2008). Generally, the visualization in lower dimensions (usually two or three) involves projecting high-dimensional item features onto a 2D plane or a 3D space, such as self-organizing maps (SOMs). These techniques have realized the active and creative music management and recommendation on the level of songs and artists.

To support exploration of unknown music, Pampalk (2001) presented the *Islands of Music* interface based on audio features. They used a metaphor of geographic maps in this intuitive interface where "islands" represented self-organized clusters of similar musical pieces. Torrens et al. (2004) used metadata of sound files without acoustic analysis to visualize musical pieces in a circle, rectangle, or tree map. They suggested using this technique to create playlists by drawing paths or selecting regions of interest. Instead of visualizing the whole collection, *Musicream* (Goto and Goto 2005) dynamically showed a part of the collection to induce active user interaction. This interface enabled users to come across unexpected musical pieces and generate a playlist of playlists. *FM4 Soundpark* (Gasser and Flexer 2009), a content-based music recommender, came into use with two user interfaces: "a more traditional Adobe Flash-based MP3 player interface with a small integrated visualization of similar tracks to the currently played one and a downloadable, fully interactive, 3D visualization client". Donaldson (2007b) visualized the prominent

[9] http://www.myspace.com/

structure of a recommendation network, and provided an interaction function to deal with the occlusion in the low-dimensional representation.

On the other hand, there are interfaces that focus on the level of artists. van Gulik et al. (2004) presented an interface of artist map, where artists were drawn as dots in a 2D space and similar artists were placed close together. It enabled users to explore and discover music collections on small devices. Based on the artist map, van Gulik and Vignoli (2005) described a visual playlist creation method by specifying paths or regions on the map. *MusicRainbow* (Pampalk and Goto 2006) mapped all artists in a collection onto a circular rainbow where colors represent different styles of music. Similar artists were mapped near each other by the traveling salesman algorithm and summarized with word labels extracted from web pages. Later, Pampalk and Goto (2007) designed the *MusicSun* user interface as an artist recommendation tool. *MusicSun* is a query-by-example interface, where the query artists were placed inside the sun encircled by associated text labels, and recommendations were displayed on the right side of the sun.

4.8 Playlist as Recommendation Engine

Music listeners typically do not listen to a single song in isolation but rather a sequence of songs (McFee and Lanckriet 2011). A playlist is a group of songs, from an unordered list to a specific ordered list, or anything in between (Knees et al. 2006; Fields 2011). To simplify, a playlist refers to any set of music selected for a particular purpose. Due to the context of selecting and ordering a set of songs from a music collection, playlist generation can serve as a particular tool for music recommendation (Pauws and Eggen 2003; Herlocker et al. 2004; Kaji et al. 2005; Cai et al. 2007; Celma 2010; Fields 2011).

In many cases, a playlist is created to fit the current situation (Cunningham et al. 2006; Knees et al. 2006; Liu et al. 2009; Fields 2011). For example, a playlist can be created as background music for another task such as studying, driving, or exercising. Or it may be created to reflect a particular mood or emotion such as depression or excitement. Different from a classic music recommendation list that usually orders musical pieces by descending similarity scores or popularity, a playlist can utilize a number of methods to order songs, including random, alphabetical, chronological, sorting acoustic attributes, optimizing song transitions and rule-based ordering (Aucouturier and Pachet 2002a; Donaldson 2007a; Fields 2011). Thus, playlist generation can solve the problem of context-aware mu-

sic recommendation with an ordered list. On the other hand, playlist generation can be different from general music recommendation in terms of orientation. While recommendation systems are classically defined about consumption and acquisition, playlists are for listening and playback (Pestoni et al. 2001; Fields 2011).

Over the last few years, a number of software applications or web services have included the playlist generation. One famous example is the functions within *iTunes*[10], such as *smart Playlists*, *Genius Playlists*, and *Genius Mixes*. The *smart Playlists* feature allows a user to specify a number of rules or constraints on various metadata (e.g., artist and genre) which are then used to create playlists automatically. The *Genius Playlists* feature makes a playlist of songs from the user's library based a song selected by the user. The *Genius Mixes* feature creates mixes by automatically grouping the user's collections. A number of web-based services are also offering personalized radio services using various analytics to generate playlists, such as *Pandora* and *Last.fm*.

The construction of a playlist is tied to the understanding of how songs relate to each other. Both the collaborative filtering and content-based filtering can be used to facilitate the creation of playlists. In the cases of Hayes and Cunningham (2000), French and Hauver (2001), Pestoni et al. (2001), and Avesani et al. (2002) collaborative filtering reveals the social relationships of music and builds playlists by finding users with similar interests. In terms of non-social playlist generation, Logan (2002, 2004), Pampalk et al. (2005), and Flexer et al. (2008) used audio similarities, while Platt et al. (2002) and Pauws and Eggen (2003) worked with some kinds of metadata (e.g., composer, instrument, and producer). Knees et al. (2006) used a combination of audio features and web-based artist similarity to accelerate the similarity computation on song level. Specially, existing playlists can support the process of similarity exploration. Ragno et al. (2005) assumed that songs that appeared in close proximity in playlists were more likely to be similar, such that they inferred similarities between songs based on their occurrences in playlists. Maillet et al. (2009) used playlists from professional radio stations as training data to learn a similarity model based on song-level audio features. The output indicated the probability of audio files being played successively in a playlist.

Given the structure of a collection of songs with respect to similarity, a simple case of playlist generation is to return a subset of similar songs as a playlist (Logan 2002; Pampalk et al. 2005; Pampalk and Gasser 2006; Fields 2011). On the other hand, some researchers

[10]http://www.apple.com/itunes/features/

paid attention to optimize a route for the sequence of songs. Alghoniemy and Tewfik (2001) treated the problem of playlist generation as a network flow problem. Based on integer linear programming and branch & bound techniques, their algorithm found a path (of user-defined length) satisfying user defined constraints and linking the given start song and end song. Flexer et al. (2008) looked for an inherent sequential order by creating a smooth transition between a start song and an end song. As a result, the songs at the beginning sounded similar to the start song, songs at the end similar to the end song and songs in the middle similar to both start and end songs. Baccigalupo and Plaza (2006) applied case-based reasoning to create new playlists with inherent temporal structure based on the patterns of existing playlists. Knees et al. (2006) used the traveling salesman algorithm to generate playlists where consecutive tracks are maximally similar on average. Liu et al. (2009) generated a playlist according to the heartbeat transition via the Markov decision process. Bosteels et al. (2009) processed the procedure of dynamic playlist generation using fuzzy set theory. Hu and Ogihara (2011) used time-series analysis of genre and year to predict the next song to play. They argued that their approach catered better to a user's taste than recommending a similar song to the current one. Fields et al. (2010) and McFee and Lanckriet (2011) discussed the evaluation of playlists in terms of the sequence prediction.

Some playlist generation systems allow users to specify additional requirements. For example, the first song should be from a particular artist, there should be 13 hip-hop songs in a playlist of 30 songs, or the playlist should contain titles with increasing tempo. These requirements can be modeled as logical constraints of music attributes (e.g., artist, genre, and tempo) over playlist positions (Pachet et al. 2000; Aucouturier and Pachet 2002a; Pauws and van de Wijdeven 2005; Pauws et al. 2006). To scale up playlist generation, the local search procedure to constraint satisfaction was proposed and realized. According to user-defined constraints, Aucouturier and Pachet (2002c) proposed a cost function of constraints and constructed the playlist by iteratively optimizing an initial randomly chosen playlist with regard to the cost function. Pauws et al. (2006) performed the local search procedure by means of simulated annealing algorithm. However, searching for an optimal sequence based on multiple constraints is an NP-hard problem (Aucouturier and Pachet 2002c). It is difficult to extend such methods to a scale with thousands of pieces and hundreds of constraints (Cai et al. 2007).

In addition to these automatic algorithms, there have been other works that utilized visualization techniques and human interaction to generate playlists. Goto and Goto (2005) presented an interactive user interface where users can grab pieces of music from similarity-based flows of tracks to create playlists. van Gulik and Vignoli (2005) allowed users to draw paths and specify regions on an artist map to construct playlists. Baur et al. (2010) presented a real-time interaction model for creating playlists on mobile devices. Beginning with a seed song, the system first delivered a set of recommendations. After the user selected one and added it into the playlist, the process repeated with the selected song as a new seed to make recommendations. In this way, the user manually constructed playlists with the assistance of automatic recommendation.

4.9 Group Recommendation

Recommendation systems have traditionally focused on recommending items to individual users (Jameson and Smyth 2007; J.-K. Kim et al. 2010). It is not a limitation in the scenarios where users behave individually such that only their personal interests should be considered. However, in some domains of recommendation, for example, music recommendation, users may be looking for music in groups, such as at parties or in the gym. In such cases, personalized music recommendation systems have difficulties in achieving a proper degree of group satisfaction. Recently in the area of music recommendation, there have been a small number of studies that address their recommendations to groups of users, varying from long-term communities established formally (J.-K. Kim et al. 2010) to short-term collections of individuals on a particular occasion (Cho et al. 2007).

Group recommendation systems differ from individual recommendation systems in that group recommendation systems need to aggregate individual users' tastes into a group preference properly (Crossen et al. 2002; Chao et al. 2005; J.-K. Kim et al. 2010). One of the earliest group recommendation systems is *MusixFX* (McCarthy and Anagnost 1998), which automatically adjusted the selection of music playing in a fitness center. The system kept a database of members' preferences for a wide range of musical genres and constructed a group model through an averaging procedure. To maximize the satisfaction of the group, it played the genre with the highest average ratings. Without explicit specification of preferences, *Flytrayp* (Crossen et al. 2002) paid attention to what music people listened to on their computers. Using this track data together with the interrelationships of genres, *Fly-*

trayp decided the next song to play based on a voting mechanism. Instead of determining users' preferred songs, *Adaptive Radio* (Chao et al. 2005) elicited negative preferences (e.g., expressions of dissatisfaction with particular music tracks). It kept track of the songs that users disliked and avoided playing them. The adaptive in-vehicle multimedia recommendation system (Yu et al. 2005) aggregated user profiles through wireless mobile devices, such as laptops, PDAs, and cell phones. The aggregation procedure first selected features to represent common interests of travelers, and then assigned appropriate weights to the selected features. J.-K. Kim et al. (2010) proposed a group recommendation system for members of online communities. The procedure of generating recommendations consisted of two phases. First, a group profile-based filtering method produced a candidate recommendation set through an adjusted collaborative filtering process. This phase aimed to satisfy the majority of group members. Second, an individual profile-based filtering method removed items irrelevant to individual preferences from the candidate set, which could reduce the dissatisfaction of individuals. Although representing the taste of the group before making recommendations can increase the chance of finding valuable recommendations, it cannot satisfy all members of the group (Jameson and Smyth 2007; J.-K. Kim et al. 2010).

Instead of aggregating individual users' preferences into a group preference, there exist other schemes to make recommendations to groups. O'Connor et al. (2002) generated recommendation sets for each group member and then merged them into a final recommendation set for the group. This method explored the tradeoffs between group satisfaction and individual privacy for small groups of close friends. However, this recommendation approach can be very time-consuming for larger groups, and may not function effectively for more anonymous groups. For public places where people are coming and going frequently, Cho et al. (2007) proposed a context-aware music recommendation system based on sensor networks. According to the information collected by the installed sensors, such as the density of people, season, weather, and time, the system chose the music that best matched the current situation from the music database.

A comprehensive description of the recommendation to groups can be found in Jameson and Smyth (2007).

Chapter 5

Evaluating music recommendation systems

Given many approaches to generate music recommendations, how to identify the best algorithm is important for both the academic research and the e-commerce application. Evaluating a proposed method is one of the biggest challenges when designing music recommendation systems (Barrington et al. 2009). This challenge mainly lies in the disagreement on the attributes to be measured and the metrics to be used (Herlocker et al. 2004). To evaluate how well a commercial recommender achieves its overall goals, one possible way is to check the revenue of the application with and without the recommendation strategy and make an estimation of the value of the system (Shani and Gunawardana 2011). Although this evaluation approach can provide conclusive evidence of the system performance, it is time-consuming because a period of time is always needed to validate the revenue. It is also not commercially viable and different vendors can not be compared. Alternative possibilities are to evaluate recommendation systems in terms of some specific properties. In the literature, the majority of previous work has focused on improving the predictive accuracy of recommendations. However, some studies have shown that users are willing to sacrifice some amount of accuracy for improved performance in other aspects, and that such systems are more satisfying overall (Swearingen and Sinha 2001; Herlocker et al. 2004; Ziegler et al. 2005; Zhang et al. 2012). Therefore, a growing trend is to consider properties other than accuracy such as coverage, novelty, and diversity, which contribute towards the quality of music recommendations. All these properties will be discussed below.

As to the problem of collecting evaluation data, Shani and Gunawardana (2011) proposed three types of experiments. Firstly, offline simulation utilizes existing datasets and requires no user interaction, thus it is easy to conduct. Secondly, an user study involves a small group of subjects who are required to perform predesigned tasks in a controlled environment. This approach can collect both the quantitative and the qualitative information, but the bias in users need to be taken into consideration. Lastly, large-scale online evaluation refers to the case where a recommendation system is used by real users performing real tasks. Since this approach collects evaluation results from real user populations interacting with the system, it is more trustworthy than offline simulation and user studies of small set of subjects (Kohavi et al. 2009; Shani and Gunawardana 2011). Nevertheless, such experiments can be expensive and risky in some cases, such that they are not easy or feasible to be carried out. There has not been any notable work on conducting an online evaluation in the academia or the industry. Here, only the description and examples of the first two experimental settings, offline simulation and user studies, will be given.

5.1 Properties

As different applications have different needs, it is crucial to decide on the properties for evaluation. Since some of the properties (e.g., novelty, diversity, and privacy) often conflict with the drive for accuracy, it is also important to understand how these traded-offs affect the overall performance.

5.1.1 Accuracy

It is assumed that a system providing more accurate predications will be preferred by users, so that many works in evaluating recommendation systems have focused on the property of predictive accuracy (Swearingen and Sinha 2001; Herlocker et al. 2004; Ziegler et al. 2005; Hu et al. 2008; Zhang et al. 2012). Accuracy may measure how well a system predicts a rating value for a specific item, how often a system makes correct or incorrect recommendations, or how close a predicted ranking of items differs from the user's true ranking.

Rating-based Metrics

In some applications, the system predicts the rating value of an item from a user on a numeric range (e.g., 1–5). In this case, the deviation between the predicted value and the actual one can be used to demonstrate the prediction accuracy of the system. Given a test set T of user-item pairs (u, i) with actual ratings $r_{u,i}$ and a set of predicted ratings $\hat{r}_{u,i}$ generated by the system, mean absolute error (MAE) (Shardanand and Maes 1995; Herlocker et al. 1999; Pennock et al. 2000; Sarwar et al. 2001; J.-H. Kim et al. 2006; Rashid et al. 2006; Li et al. 2007; Liu et al. 2009) and root mean squared error (RMSE) (Amatriain et al. 2009; Dror et al. 2011) are commonly used metrics.

$$MAE = \frac{1}{|T|} \sum_{(u,i) \in T} |\hat{r}_{u,i} - r_{u,i}| \tag{5.1}$$

$$RMSE = \sqrt{\frac{1}{|T|} \sum_{(u,i) \in T} (\hat{r}_{u,i} - r_{u,i})^2} \tag{5.2}$$

Decision-based Metrics

Some applications are not interested in how much a user likes recommended items (represented as rating values), but rather focus on whether or not the user would like to accept the recommendations. For instance, a system may recommend a set of songs to a user and the user may add them into a playlist or not. A contingency table (or confusion matrix) as shown in Table 5.1, shows four possible outcomes, which is helpful to measure the appropriateness of recommendations.

Table 5.1: A contingency table showing possible results of recommendations

	Liked	Not liked
Recommended	True Positive (TP)	False Positive (FP)
Not recommended	False Negative (FN)	True Negative (TN)

TP and TN are desired outcomes when the relevant items are recommended and the irrelevant items are ignored. FNs describe the cases when products are not recommended, though the consumer would like them, while FPs represent the cases when products are

recommended, though the consumer does not like them. In many cases, such as medical diagnosis or news recommendation, FNs are more important than FPs because the cost of an FN is significant. An FN in diagnosis may be at the expense of a patient's life and an FN in news recommendation is likely to lead to an investment failure. Differently, in music or movie recommendation, the cost of an FN is almost zero. According to Sarwar et al. (2000), FPs are the most important errors that need to be avoided for they will lead to angry consumers. However, in the area of music recommendation, the cost of an FP is just a few dollars to purchase a song or an album, although FP is more significant than FN.

In order to quantify the performance of a recommendation system, precision and recall can be utilized (Hijikata et al. 2006; Cai et al. 2007; Lee and Lee 2007; Bozzon et al. 2008; Fields et al. 2010).

$$Precision = \frac{TP}{TP + FP} \tag{5.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{5.4}$$

Precision measures the fraction of recommended items that are relevant, while recall measures the fraction of relevant items that are recommended (Karypis 2001; Bozzon et al. 2008; Celma 2010). Recall can also be viewed as the probability of a randomly-selected good item being recommended. A recall value of 1.0 indicates that the recommendation algorithm is always able to recommend the good items, whereas a recall value of 0.0 indicates that the recommendation algorithm is not able to recommend any of the good items (Karypis 2001). In a typical online music recommendation system (e.g., *Last.fm*), users require a higher precision to receive highly relevant recommendations; while, in an offline system (based on browsing and navigation activities), users prefer a higher recall in order to retrieve a wider range of items (Bozzon et al. 2008). As a weighted harmonic mean between precision and recall, F-measure, F_β, has also been utilized for performance evaluation (Hijikata et al. 2006; Bozzon et al. 2008; Celma 2010).

$$F_\beta = \frac{(1 + \beta^2) \cdot (precision \cdot recall)}{\beta^2 \cdot precision + recall}. \tag{5.5}$$

Taking the ranking performance into consideration, Bozzon et al. (2008) adopted the precision-recall curve, where recall was plotted on x-axis and precision was plotted on y-axis. This curve indicated the precision achieved by retrieving the top-k% relevant items in the collection. An alternative tool is the receiver operating characteristic (ROC) curve

that measures the selection of high-quality items over a range of different recommendation list lengths (Celma 2010; Shani and Gunawardana 2011) or over a scope of prediction score thresholds for recommendation (Herlocker et al. 1999; Su and Khoshgoftaar 2009). On an ROC curve, false positive rate (FPR) is plotted on the x-axis and true positive rate (TPR) is plotted on the y-axis.

$$FPR = \frac{FP}{FP + TN} \tag{5.6}$$

$$TPR = Recall = \frac{TP}{TP + FN} \tag{5.7}$$

The FPR can be used to determine the probability of a randomly-selected bad item being rejected by the recommender (Herlocker et al. 1999). The trade-off between FPR and TPR can be visualized by the ROC curve. Typically, the area under the curve (AUC) can summarize the ROC results and compare algorithms independently of applications (Schein et al. 2002; Celma 2010). The AUC demonstrates the probability that a system will be able to choose correctly between a good item and a bad item (Herlocker et al. 2004). A higher AUC value infers a better performance of the recommendation system.

Ranking-based Metrics

Sometimes the application presents ranked lists of recommendations to users with the interest in ordering items according to users' preferences. Ranking-based accuracy metrics can assist in evaluating a recommendation sequence, such as an ordered playlist, although they may be overly sensitive for applications where users are just concerned about whether an item is good or not (Herlocker et al. 2004).

There are two common approaches to evaluate the accuracy of such a ranked list. The first one needs to generate the correct order of a list for a given user, and compare the predicted list with the correct one. The second approach needs to measure the utility of the ranked list to the user, based on the assumption that the top positions are more relevant than the bottom positions on a list (Shani and Gunawardana 2011). In other words, the utility-based ranking metrics take the item position on a predicted list into consideration, which include mean reciprocal rank (MRR) (Nanopoulos et al. 2010), discounted cumulated gain (DCG) (Firan et al. 2007; Cantador et al. 2010), and rank score (R-score) (Breese et al. 1998; J.-H. Kim et al. 2006; Li et al. 2007; Yoshii and Goto 2009).

The MRR evaluates the quality of top-N lists generated by retrieval processes by measuring how far from the top appears the first good item. It is defined as

$$MRR = \frac{1}{N}\sum_{i=1}^{H}\frac{1}{p_i}, \tag{5.8}$$

where H is the number of hits that refer to the good recommendations, and p_i is the position of hits within the top-N list. Hits that occur earlier in the top-N list are weighted higher than hits that occur later in the list.

The DCG logarithmically penalizes relevant recommendations that are located in the bottom of the top-N list. Assuming each user has a gain, G, from being recommended an item, the DCG at position p in an ordered list can be formulated as,

$$DCG_p = \begin{cases} G_1 & \text{if } p = 1 \\ DCG_{p-1} + G_p/\log(p) & \text{Otherwise.} \end{cases} \tag{5.9}$$

Therefore, the utility of a top-N recommendation list for a user is DCG_N that is based on the user's perceived gain.

The R-score assumes that the value of recommendations decline exponentially down the ranked list. The strength of the decline is described by a half-life parameter α, such that R-score is also called half-life utility. The half-life is the number of items on a list that have a 50% chance of being listened to. The R-score for each user is defined as,

$$R_u = \sum_p \frac{\max(r_{u,p} - d, 0)}{2^{(p-1)/(\alpha-1)}}, \tag{5.10}$$

where $r_{u,p}$ represents the rating of user u on the item at position p in the list, and d is a task dependent neutral rating value. In the literature, d can be a default rating (Yoshii and Goto 2009; Celma 2010), the overall mean rating (Li et al. 2007), or the mid-average rating (J.-H. Kim et al. 2006). The total score reflecting the utilities of all users is,

$$R = 100\frac{\sum_u R_u}{\sum_u R_u^{max}}, \tag{5.11}$$

where R_u^{max} is the maximum achievable utility if all observed items had been at the top of the ranked list, ordered by rating values. A higher rank score indicates not only

higher accuracy, but also higher probability of recommending known items (Yoshii and Goto 2009).

5.1.2 Utility

The measurement of users' perceived utility requires users to check the recommended items and explicitly express their opinions over them (Herlocker et al. 2004). An obvious procedure is to run a user study and collect preference data through an interview or using a questionnaire. For collaborative filtering systems, the ratings can be used to measure the utility (Breese et al. 1998). For example, in the case of rating an album, a five star rating indicates higher utility than a four star rating. Note that a user's expression of utility may be biased or imprecise (Barrington et al. 2009). This is because the personal sentiments and contexts may affect the acceptance of recommendations. To measure these effects, Park and Moon (2011) applied a conceptual model called technology acceptance model, which had been used to explain users' acceptance behaviors in socio-psychology studies. Moreover, aggregating the opinions across users into one score for a recommendation system needs more consideration. It is not clear whether to treat all users equally or not. In the e-commerce, the opinions of users who bought many items may be preferred by the system. Meanwhile, these users' preferences to this system may be higher than the users who only bought a single item (Shani and Gunawardana 2011). Therefore, assigning proper weightings to users may be not easy.

5.1.3 Confidence

Confidence in recommendations can refer to the system's trust in its own recommendations or represent users' trust in the recommendations (Shani and Gunawardana 2011). The goal of a recommendation system as a decision-support system is to help users make the best possible decision about what to try or purchase based on their interests (Swearingen and Sinha 2001; Herlocker et al. 2000). In many cases, the system confidence can help users make effective decisions. For example, if the system recommends an album with very high confidence, and another album with the same rating but a lower confidence, the user is more likely to choose the one with higher confidence. A common measurement of system confidence is the probability that the predicted value is indeed true, or the interval around the predicted value where a predefined portion (e.g., 95% of the true values) lie. Herlocker

et al. (2000) explored a wide range of different system confidence in order to study their influences on users.

On the other hand, the measurement of users' trust generally requires the involvement of users to provide feedback. Some studies have showed that presenting a recommendation that a user love or hate a lot can significantly affect the overall evaluation (Cunningham et al. 2006; Lee 2011). For example, a user may rate a playlist higher because he/she really likes one of the songs, or rate the whole playlist lower if he/she dislike a single song (Lee 2011). Another notable pattern is that users seem to like rediscovering songs that were once familiar (Swearingen and Sinha 2001; Lee 2011). Recommending some familiar items to users is deemed to be able to improve users' confidence in the system, even though users gain little value from them (Celma and Herrera 2008). If users receive unknown recommendations, the system is better to provide transparency by giving an explanation of why the recommendations are produced. Herlocker et al. (2000) showed that providing more transparency can increase users' trust because the recommendations sound reasonable. Generally speaking, given two recommenders that perform similarly on other properties, the system with higher (system or user) confidence may be preferred (Shani and Gunawardana 2011).

5.1.4 Coverage

In the item space, coverage measures the domain over which the system can form predictions or make recommendations (Herlocker et al. 2004). Systems with lower item coverage may be less valuable to users because they will miss the items from neglected area which match users' preferences (Seyerlehner et al. 2009; Celma 2010). Items out of coverage can never be annotated or rated by users either. A practical measurement of coverage is the percentage of the items for which predictions can be formed. Alternatively, the catalog coverage is defined as the percentage of items in the catalog that are ever recommended to users, probably in an experiment. Most of the time, the item coverage should be considered together with accuracy, for the fact that the items uninteresting to all users are better to be filtered out for higher accuracy although leading to lower coverage (Shani and Gunawardana 2011).

In the user space, coverage is the proportion of users or user interactions to whom the system can make recommendations (Shani and Gunawardana 2011). In some cases, the system may not recommend items to users if the system has little information about them, or the users' preferences are too obscure to find matched items. Specifically in collaborative

filtering, the user coverage can be measured as the number of ratings a user has to provide before receiving recommendations (Swearingen and Sinha 2001).

5.1.5 Novelty and Serendipity

Novel recommendations are those providing unknown but relevant items to users (Lee 2011). The novelty can be defined as the margin with respect to user's knowledge and degree of interest in a recommended item (Yang and Li 2005). Serendipitous recommendations offer users the opportunity to find surprisingly interesting items that they might not have discovered (Herlocker et al. 2004). Unlike novelty, serendipity can be imagined as the distance between the recommended item and the user's expected content (Zhang et al. 2012). To clarify the difference between novelty and serendipity, consider the situation of recommending music to a user who has listened a number of songs from Beatles. Recommending other songs from Beatles may be novel to the user, if he/she is unaware of those songs, but not serendipitous. But recommending music from an obscure garage band is more likely to be serendipitous. In short, serendipitous recommendations are by definition also novel but novel recommendations do not have to be serendipitous. Sometimes serendipity is incorrectly used the same as novelty (Herlocker et al. 2004; McNee et al. 2006), because they both measure the nonobviousness of a recommendation. However, the distinction between them is important when evaluating collaborative filtering algorithms. Traditional content-based filtering systems do not have the potential for serendipitous recommendations, since they are always looking for items similar in content.

An obvious way to measure novelty is to gather feedback from users explicitly or implicitly (McNee et al. 2006). Explicit feedback is based on two related questions: whether the user already knew the item, and whether he/she likes it or not. Implicit feedback includes, for instance, the purchase or preview of an item. If a user already knew an item, preview and repeat purchase are less likely to happen. Alternatively, objective metrics of novelty can make use of the item popularity. It seems that popular items are more likely to be recognized and rated, and items with fewer interactions (e.g., rating or purchasing) are more likely to be unknown to users (Ziegler et al. 2005; Zhang et al. 2012). Under this assumption, the novelty is inversely proportional to the popularity of the recommended items. That is, lower score of popularity denotes higher novelty. Celma and Herrera (2008) correlated the Long Tail distribution and complex networks to assist in the novelty analysis. To measure the potential of serendipity, the distance between the recommendations

and the items in the user's profile has been utilized (Zhang et al. 2012). Higher values indicate that recommendations deviate from a user's traditional tastes, and hence are more surprising. Since a serendipitous system challenges users to expand their tastes, hopefully it will provide more interesting recommendations that can help improve users' satisfaction (Swearingen and Sinha 2001).

However, novelty and serendipity may contradict the desire of accuracy, although they are sometimes necessary to improve users' experience in discovery (Celma 2010; Lee 2011). For example, recommending the Beatles' famous White Album to a Beatles fan is accurate but neither novel nor serendipitous, while recommending music from an obscure artist can be novel and serendipitous but not accurate. Therefore, when evaluating how well a recommendation system can make a user aware of previously unknown items, it is worthwhile to check to see to what extent users accept the new recommendations (Herlocker et al. 2004). Users may at first be intrigued to try out the unexpected recommendations, but stop following them if they find recommendations are not satisfying. A user-centric method proposed by Celma and Cano (2008) aimed at measuring users' perceived quality of novel recommendations. Although it is not clear about the ideal proportion of familiar and new songs in a recommendation list, users do prefer a mix of familiarity and novelty/serendipity (Lee 2011). Zhang et al. (2012) suggested that a personalized recommendation system should be able to allow users to individually tune the level of desired novelty and serendipity.

5.1.6 Diversity

Diversity typically measures how many different items are present in a list of recommendations (Smyth and McClave 2001; Fleder and Hosanagar 2007; Zhang et al. 2012). Since users may listen to music repeatedly and continuously, diversity is deemed important for a music recommendation system to generate a broader range of relevant choices and hopefully to increase users' satisfaction (Slaney and White 2006). In music, diversity may refer to various attributes, such as genre, artist, instrumentation, lyrics, tempo, and mood (Lee 2011). Diversity comes at the expense of homogeneity or coherence of a recommendation list, and sometimes it even counteracts the accuracy (Smyth and McClave 2001; Ziegler et al. 2005; Zhang et al. 2012). For example, a playlist of Beatles is coherent and accurate for a rock music fan, but seems monotone for a user with multiple interests. On the other hand, a playlist consisting of multiple genres of music is diverse but not accurate enough for a big rock music fan.

The item-to-item similarity has been well explored to measure diversity (Shani and Gunawardana 2011). If the items on a list are all similar to each other, the diversity will be low. One possible definition of diversity is the average dissimilarity between all pairs of items in an item set (Smyth and McClave 2001; Zhang and Hurley 2008). Alternatively, the intra-list similarity (ILS) metric introduced by Ziegler et al. (2005) sums the pairwise similarity of all items in a set. A list of dispersed and diverse recommendations scores a lower ILS value than the list with similar items (Zhang et al. 2012). Slaney and White (2006) characterized the diversity of a playlist by fitting a Gaussian probability model to the songs. For each song, a set of audio features were calculated to place that song as a point in a multidimensional space. The volume of an enclosing ellipsoid generated by the model was used to describe the diversity.

5.1.7 Privacy

A personalized recommendation algorithm requires user information in order to make recommendations. In general, the more information the system knows about its users, the better recommendation service the users will be able to receive (Resnick and Varian 1997). Gathering more user information potentially increases recommendation accuracy, but also increases the risk of privacy issues (Lam et al. 2006; Shani and Gunawardana 2011). Privacy problems may arise when people who are observed are not aware of it. For example, *Netflix* will be paying out millions of dollars to settle a lawsuit that accused the company of keeping customer data of canceled subscribers for longer than they should have kept it. Thus, a personalized system needs to be careful about the acquisition, storage and application of user information.

The challenge for the adoption of personalized services is to find a balance where the system is able to make good recommendations to users while not violating or threatening their privacy (Perik et al. 2004; Lam et al. 2006). One intuitive question is how much data is needed from a user to make good recommendations to that user. Lam et al. (2006) argued that "providing a recommender with data may produce diminishing returns". That is, once the system knows a certain amount information about a user, obtaining further information is only marginally useful. Moreover, users will be less likely to trust a system that continuously records sensitive information about their interests or tastes (Perik et al. 2004). Although the preferences for music are considered less personal than the information

about personality traits, the preference data may allow undesired re-identification (Lam et al. 2006).

5.2 Experimental Settings

When designing an evaluation experiment, it is important to be aware of several key decisions regarding experimental datasets and expenses. Can the evaluation be carried out offline on an existing dataset or does it require user subjects? How long does it take for an evaluation, hours, days, or months? In the following, the pros and cons of two experimental settings for evaluation will be discussed.

5.2.1 Offline Simulation

Offline simulation tests recommendation algorithms on an existing dataset, without interaction with real users. The assumption is that the user behavior when the data was collected is similar to the user behavior when evaluation is deployed (Shani and Gunawardana 2011). Such evaluation is attractive for the fact that it is economical to conduct, and that it allows to compare a wide range of algorithms on several different datasets at once. When a dataset is available, the evaluation process simply runs the test algorithm and compares the predicted results to the "ground truth" from the dataset. The data used for the offline evaluation should be as close as possible to the data that the recommendation system will face when deployed online (Herlocker et al. 2004). The limitation of offline simulation lies in the incompleteness of dataset. It is not possible to evaluate the appropriateness of a recommended item to a user if no information about that user-item pair exists in the dataset. Since this evaluation does not recruit any users, only objective evaluation can be provided. Recently, there have been two offline competitions aiming to evaluate music recommendation systems.

Yahoo! Music Dataset and KDD Cup Contest

The *Yahoo! Music Dataset*[1] was released with the KDD Cup 2011 contest, which challenged the community to identify user tastes in music (Dror et al. 2011; Koenigstein et al. 2011). The dataset comprises over 260 million ratings (scores from 0 to 100) of 624,961 music items

[1] http://kddcup.yahoo.com/datasets.php

(a mixture of songs, albums, artists, and genres) by one million users over the last decade (1999-2010). All users and items are represented by opaque identifiers with the provided artist/genre/album/track taxonomy. Each item and each user have at least 20 ratings in the whole dataset. The one-minute resolution timestamps included in the ratings allow refined temporal analysis. This dataset is useful as a benchmark for content-agnostic collaborative filtering algorithms, because the anonymity of users and items makes it impossible to integrate various metadata and content analysis into the recommendation task (McFee et al. 2012).

The KDD Cup contest offered two different tasks. Task1 required predicting users' ratings of musical items. For each user, it provided at least 10 ratings in the training data, four ratings in the validation data, and the last six ratings for test. The evaluation criterion was the root mean square error (RMSE) between the predicted ratings and the true ones. Task2 was less conventional. Once again, each user and item have at least 20 ratings, out of which six are used as test data (while the rest as training data). The test data for each user contained three items rated highly (score 80 or higher) by the user and three that had not been rated. With the goal to differentiate high ratings from missing ones, participants were asked to identify exactly the three highly rated items for each user. The evaluation criterion was the error rate, the fraction of wrong predictions. At both tasks, the test set was internally split into two equal halves known as Test1 and Test2 (Dror et al. 2011).

Observed from the results, all the top 3 winning teams have a slightly better performance on Test2 than on Test1 for both Task1 and Task2. Only the results on Test1 will be quoted here for illustration. On Test1, the best result achieved on Task1 was RMSE of 21.01 (Chen et al. 2011), and the best result on Task2 was an error rate of 2.47% (McKenzie et al. 2011). Both were achieved by the National Taiwan University team. For Task1, Chen et al. adopted a four-stage procedure: individual model building, non-linear blending, linear ensemble, and post-processing. The first step of individual model building included variants of existing models, such as matrix factorization, k-nearest neighbors, probabilistic latent semantic analysis, probabilistic principle component analysis, supervised regression, and restricted Boltzmann machine. Step 2 blended these individual models in a non-linear manner, and then step 3 combined both the individual and the blended models through a linear ensemble. Finally, the post-processing step adjusted the predictions generated from the ensemble model. McKenzie et al. utilized a similar approach of local blending and global ensemble to achieve accuracy over 97%. The Commendo team that won the second

place in Task1 and the third place in Task2, also made use of an ensemble, blending many solutions. Their achievement on Test1 for Task1 was RMSE=21.08, and the error rate on Test1 for Task2 was 2.49%. The prevalence of ensemble learning reinforced the finding from the *Netflix* experience.

Note that the RMSE results of Task1 are not directly comparable to those of the *Netflix* contest, because they are using different rating scales. While in the *Netflix* dataset the rating range is 4 (ratings from 1 to 5), the *Yahoo! Music Dataset* has a rating range of 100. Calibrating *Yahoo!* ratings to the *Netflix* rating scale will shrink the RMSE with a factor of 25. That is, the best RMSE achieved at Task1 will be around 0.84, strikingly close to the best score known on the Netflix dataset (0.86) (Dror et al. 2011).

Million Song Dataset Challenge

As indicated by its name, the *Million Song Dataset*[2] contains a collection of a million contemporary popular songs (Bertin-Mahieux et al. 2011; McFee et al. 2012). Since this dataset is linked to several complementary datasets (e.g., EchoNest and MusicBrainz[3]), it contains extensive metadata, audio features, tags on the artist- and song-level, lyrics, and so on. The collection of data used in the competition is known as the *Taste Profile Subset*[4], which consists of more than 48 million triplets $(user, song, count)$ gathered from user listening histories. The data involves approximately 1.2 million users and more than 380,000 songs, where all users have at least 10 songs in profile. Due to user privacy concerns, users are anonymous. Unlike the *Yahoo! Music Dataset* which contains explicit ratings, this dataset uses playcounts instead. Although having played a track does not mean it was liked, the fact a song was listened to suggests that there was at least some initial interest on behalf of the user. Another difference is that timestamps are not provided in the dataset (McFee et al. 2012).

The *Million Song Dataset Challenge* aims to provide a large-scale, transparent music recommendation challenge allowing participants to exploit metadata and content analysis. In the contest, the recommender observes a subset of the songs consumed by a user, and predicts a ranking over all other songs in the dataset. Ideally, the remaining songs consumed by the user would be ranked ahead of all songs not consumed. During the evaluation, all

[2]http://labrosa.ee.columbia.edu/millionsong/
[3]http://musicbrainz.org/
[4]http://labrosa.ee.columbia.edu/millionsong/tasteprofile

that matters is whether the user listened to the song or not, rather than the exact number of times the song was consumed. Due to the large scale of the test data (100K users and 380K songs), only the top-500 predicted rankings are evaluated. The evaluation procedure uses the mean average precision of the truncated ranking as the primary evaluation metric and also includes additional metrics such as mean reciprocal rank, normalized discounted cumulative gain, precision-at-10, and recall at the cutoff point (McFee et al. 2012). The preliminary results released online reveal the leaderboard[5] ordered by the mean average precision, which still needs further verification before finalization.

5.2.2 User Study

A user study regularly recruits a group of subjects to interact with recommendation systems. Quantitative measurement can be collected during subjects performing tasks through observing their behavior, such as how many recommendations are accepted by the user or the ratings indicating how much the user likes the recommended items. On the other hand, qualitative measurement can be collected via questionnaires or surveys. The subjects may be required to answer questions such as whether they have received interesting recommendations, or whether they trust the system (Shani and Gunawardana 2011).

Unlike offline simulation, user studies allow studying user behavior through subjects' interaction with the system and collecting qualitative data that is often crucial for estimating the recommendation performance (Hu and Ogihara 2011). However, it is not assured that users will behave in the same way in real life as in the lab (Swearingen and Sinha 2001). The results can be biased because subjects are aware that they are participating in an experiment. If the subjects know the hypothesis tested, they may tend to support it and satisfy the conductor of the experiment. Since music is in most cases embedded into a socio-cultural process, Baumann et al. (2004) suggested carrying out user studies outside the lab during daily activities. Moreover, this setting will be expensive and time-consuming if a large set of subjects are needed. Therefore, only a small set of subjects and relatively short tests can be recruited in a user study.

In the literature about music recommendation, the group size of many user studies was smaller than 25 subjects. For example, Lee (2011) involved eight participants to examine the perceived quality of generated playlists and argued the importance of factors other than similarity, such as variety, personal preference, familiarity, and mix of familiar and

[5] http://www.kaggle.com/c/msdchallenge/leaderboard

new music. Chen and Chen (2005) invited 10 students to test the performance of their proposed systems by measuring precision. Knees et al. (2006) carried out a user study of 10 subjects to score the consistency of the playlists generated by their approach. In Bogdanov et al. (2010), 12 people participated in subjective listening tests and then expressed different subjective impressions related to the recommended music, with regards to familiarity, enjoyment, and listening intention. The evaluation results revealed that their music recommender based on semantic similarities was preferred by users over the timbre-based recommender; however, it was inferior to the considered collaborative filtering recommender in terms of both the number of successful novel recommendations and the trusted recommendations. Magno and Sable (2008) compared their proposed recommendation engines to today's leading online music discovery tools (i.e., *Pandora*, *Last.fm*, and *Allmusic*), based on a user study of 15 volunteers. The analysis of the user ratings assigned to recommendations, showed that a signal-based recommendation engine could perform comparably to popular commercial music discovery applications when subjected to human evaluation. The results also showed that music recommendations given by an expert could not always satisfy the sensibilities of a music consumer. Pauws and Eggen (2003) employed 20 subjects to measure playlist quality by precision, coverage, and a rating score. After the experiment, an interview was used to yield supplementary findings on perceived usefulness. Zhang et al. (2012) conducted a user study involving 21 participants to assess the objective qualities of enjoyment, novelty, serendipity, and overall user satisfaction of playlists. The analysis of survey results showed that introducing serendipity into music recommendation, although sacrificed some accuracy, could improve the overall satisfaction.

According to Shani and Gunawardana (2011), the small size of user studies may cause the evaluation not to be convincing. The user studies with a relatively larger group size have also been carried out by researchers. For example, Barrington et al. (2009) employed 185 subjects to compare *Genius* to another two music recommender systems: one based purely on artist similarity and the other purely on acoustic similarity. Their three findings were: a) *Genius* had the best overall performance; b) collaborative filtering could actually capture similarities between the acoustic content of songs; c) artist similarity could account for the performance of *Genius*, when evaluators can see the names of the recommended songs and artists. Another example was the user-centric evaluation for novelty conducted by Celma and Herrera (2008), which involved 288 users and 5,573 tracks. This experiment

showed that collaborative filtering outperformed content-based methods in terms of users' perceived quality even though it recommended less novel items.

Chapter 6

Conclusion

Music recommendation systems have made progress over the last decade when numerous recommendation techniques were proposed and several "industrial-strength" systems have been developed. The recommendation for music is different from those for books and movies, due to its low cost per item, short consumption time, high per-item reuse, highly contextual usage, and numerous item types. Understanding the patterns of music listening and consumption is helpful to create accurate and satisfying music recommendations. In the literature, traditional music recommendation systems can be classified as one of two major kinds: collaborative filtering and content-based filtering. Since collaborative filtering works on the user preference data, collaborative filtering systems can follow the social consumption patterns to make recommendations, but they are prone to popularity bias and hardly ever explore the Long Tail that contain interesting and novel items for users. On the other hand, content-based filtering bases its recommendations in the content analysis of items, such that content-based systems can recommend niche items on the Long Tail but they can only recommend items similar to those already accessed. Recently, the research community has broadened its attention to include other aspects, such as hybrid approaches, user modeling, context awareness, social tagging, recommendation in the Long Tail, music networks, visualization, playlist generation, and group recommendation. So far, collaborative filtering has been popular in both commercial applications and academia, while the majority of content-based filtering methods are still at the research stage. More and more music recommendation systems have been focused on the context awareness that can provide more accurate recommendations and more satisfying services.

How to determine the best music recommendation algorithm is an important step in both the research attempt and the application design. Many evaluation works in the past have focused on improving the predictive accuracy of recommendation. However, a growing trend is to consider properties other than accuracy such as coverage, novelty, and diversity, which contribute towards the quality of music recommendations. Although it is not clear how to aggregate these properties into one score for a music recommendation system, studies do show that users are willing to sacrifice some amount of accuracy for improved performance in other aspects, and that such systems are more satisfying overall. A set of properties that are sometimes discussed as important for the music recommendation systems have been presented. These properties can be measured in either an offline simulation or a user study. An offline simulation is simple and easy to conduct, however, it can be limited by the incompleteness of dataset and can only collect the quantitative measurement. On the other hand, a user study can collect both the quantitative and qualitative information, but it is expensive to carry out such that only a small set of subjects and a relatively short tests can be recruited. The small size of user studies may cause the evaluation not to be convincing. To indicate the power of each experimental setting and harvest available experience, examples of previous work are discussed.

Bibliography

Adomavicius, G., and A. Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering* 17 (6): 734–49.

Adomavicius, G., and A. Tuzhilin. 2011. Context-aware recommender systems. In *Recommender Systems Handbook*, 217–53.

Albert, R., and A.-L. Barabási. 2002. Statistical mechanics of complex networks. *Reviews of modern physics* 74 (1): 47–97.

Alghoniemy, M., and A. Tewfik. 2001. A network flow model for playlist generation. In *Proceedings of IEEE International Conference on Multimedia and Expo*.

Alvira, M., J. Paris, and R. Rifkin. 2001. The audiomomma music recommendation system. Technical report, Massachussets Institute of Technology.

Amatriain, X., J. M. Pujol, N. Tintarev, and N. Oliver. 2009. Rate it again: Increasing recommendation accuracy by user re-rating. In *Proceedings of 3rd ACM Conference on Recommender Systems*, 173–80.

Anderson, C. 2004. The long tail. *Wired Magazine* 12 (10): 170–7.

Anderson, M., M. Ball, H. Boley, S. Greene, N. Howse, D. Lemire, and S. McGrath. 2003. RACOFI: A rule-applying collaborative filtering system. In *Proceedings of International Workshop on Collaboration Agents: Autonomous Agents for Collaborative Environments*.

Angeles, B., C. McKay, and I. Fujinaga. 2010. Discovering metadata inconsistencies. In *Proceedings of International Society for Music Information Retrieval Conference*, 195–200.

Anglade, A., M. Tiemann, and F. Vignoli. 2007a. Complex-network theoretic clustering for identifying groups of similar listeners in P2P systems. In *Proceedings of ACM Conference on Recommender Systems*, 41–8.

Anglade, A., M. Tiemann, and F. Vignoli. 2007b. Virtual communities for creating shared music channels. In *Proceedings of International Society for Music Information Retrieval Conference*, 95–100.

Aucouturier, J.-J., and F. Pachet. 2002a. Finding songs that sound the same. In *Proceedings of IEEE Benelux Workshop on Model based Processing and Coding of Audio*.

Aucouturier, J.-J., and F. Pachet. 2002b. Music similarity measures: What's the use? In *Proceedings of International Society for Music Information Retrieval Conference*.

Aucouturier, J.-J., and F. Pachet. 2002c. Scaling up music playlist generation. In *Proceedings of IEEE International Conference on Multimedia and Expo*.

Aucouturier, J.-J., and F. Pachet. 2004. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences* 1 (1): 1–13.

Avesani, P., P. Massa, M. Nori, and A. Susi. 2002. Collaborative radio community. In *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, Volume 2347, 462–5. Springer.

Baccigalupo, C., and E. Plaza. 2006. Case-based sequential ordering of songs for playlist recommendation. In *Advances in Case-Based Reasoning*, Volume 4106 of *Lecture Notes in Computer Science*, 286–300.

Balabanović, M., and Y. Shoham. 1997. Fab: Content-based, collaborative recommendation. *Communications of the ACM* 40 (3): 66–72.

Baltrunas, L., M. Kaminskas, F. Ricci, L. Rokach, B. Shapira, and K. Luke. 2010. Best usage context prediction for music tracks. In *Proceedings of 2nd Workshop on Context-Aware Recommender Systems*.

Barrington, L., R. Oda, and G. R. G. Lanckriet. 2009. Smarter than genius? human evaluation of music recommender systems. In *Proceedings of International Society for Music Information Retrieval Conference*, 357–62.

Baumann, S., and J. Halloran. 2004. An ecological approach to multimodal subjective music similarity perception. In *Proceedings of Conference in Interdisciplinary Musicology*, Volume 15, 18.

Baumann, S., and O. Hummel. 2003. Using cultural metadata for artist recommendations. In *Proceedings of 1st International Conference on Web Delivering of Music*.

Baumann, S., and O. Hummel. 2005. Enhancing music recommendation algorithms using cultural metadata. *Journal of New Music Research* 34 (2): 161–72.

Baumann, S., T. Pohle, and S. Vembu. 2004. Towards a socio-cultural compatibility of MIR systems. In *Proceedings of International Society for Music Information Retrieval Conference*.

Baur, D., S. Boring, and A. Butz. 2010. Rush: Repeated recommendations on mobile devices. In *Proceedings of 15th International Conference on Intelligent user interfaces*, 91–100.

Berenzweig, A., B. Logan, D. P. W. Ellis, and B. P. W. Whitman. 2004. A large-scale evaluation of acoustic and subjective music-similarity measures. *Computer Music Journal* 28 (2): 63–76.

Bertin-Mahieux, T., D. P. W. Ellis, B. Whitman, and P. Lamere. 2011. The million song dataset. In *Proceedings of International Society for Music Information Retrieval Conference*, 591–6.

Billsus, D., and M. J. Pazzani. 1998. Learning collaborative information filters. In *Proceedings of 15th International Conference on Machine Learning*, 46–54.

Bogdanov, D., M. Haro, F. Fuhrmann, E. Gómez, and P. Herrera. 2010. Content-based music recommendation based on user preference examples. In *Proceedings of 4th ACM Conference on Recommender Systems. Workshop on Music Recommendation and Discovery*, Barcelona, Spain.

Bogdanov, D., and P. Herrera. 2011. How much metadata do we need in music recommendation? a subjective evaluation using preference sets. In *Proceedings of International Society for Music Information Retrieval Conference*, 97–102.

Bosteels, K., E. Pampalk, and E. E. Kerre. 2009. Evaluating and analysing dynamic playlist generation heuristics using radio logs and fuzzy set theory. In *Proceedings of International Society for Music Information Retrieval Conference*, 351–6.

Bozzon, A., G. Prandi, G. Valenzise, and M. Tagliasacchi. 2008. A music recommendation system based on semantic audio segments similarity. In *Proceedings of IASTED International Conference on Internet and Multimedia Systems and Applications*, 182–7.

Breese, J. S., D. Heckerman, and C. Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of 14th Conference on Uncertainty in Artificial Intelligence*, 43–52.

Brynjolfsson, E., Y. J. Hu, and M. D. Smith. 2006. From niches to riches: The anatomy of the long tail. *Sloan Management Review* 47 (2): 67–71.

Bu, J., S. Tan, C. Chen, C. Wang, H. Wu, L. Zhang, and X. He. 2010. Music recommendation by unified hypergraph: Combining social media information and music content. In *Proceedings of International Conference on Multimedia*, 391–400.

Buldú, J., P. Cano, M. Koppenberger, J. Almendral, and S. Boccaletti. 2007. The complex network of musical tastes. *New Journal of Physics* 9 (6): 172.

Burke, R. 2002. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction* 12 (4): 331–70.

Cai, R., C. Zhang, C. Wang, L. Zhang, and W.-Y. Ma. 2007. MusicSense: Contextual music recommendation using emotional allocation modeling. In *Proceedings of 15th International Conference on Multimedia*, 553–6.

Cai, R., C. Zhang, L. Zhang, and W.-Y. Ma. 2007. Scalable music recommendation by search. In *Proceedings of 15th International Conference on Multimedia*, 1065–74.

Cano, P., O. Celma, M. Koppenberger, and J. M. Buldu. 2005. The topology of music recommendation networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 16.

Cano, P., M. Koppenberger, and N. Wack. 2005. An industrial-strength content-based music recommendation system. In *Proceedings of 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 673.

Cantador, I., A. Bellogín, and D. Vallet. 2010. Content-based recommendation in social tagging systems. In *Proceedings of 4th ACM Conference on Recommender Systems*, 237–40.

Casey, M., R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. 2008. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE* 96 (4): 668–96.

Çataltepe, Z., and B. Altinel. 2007. Hybrid music recommendation based on different dimensions of audio content and an entropy measure. In *Proceedings of European Signal Processing Conference*, 936–40.

Cebrián, T., M. Planagumà, P. Villegas, and X. Amatriain. 2010. Music recommendations with temporal context awareness. In *Proceedings of 4th ACM Conference on Recommender Systems*, 349–52.

Celma, O. 2006. Foafing the music: Bridging the semantic gap in music recommendation. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo (Eds.), *Proceedings of 5th International Semantic Web Conference*, Volume 4273, 927–34.

Celma, Ò. 2010. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. New York: Springer-Verlag.

Celma, O., and P. Cano. 2008. From hits to niches?: Or how popular artists can bias music recommendation and discovery. In *Proceedings of 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, 1–8.

Celma, O., and P. Herrera. 2008. A new approach to evaluating novel recommendations. In *Proceedings of ACM conference on Recommender systems*, 179–86.

Celma, O., and P. Lamere. 2011. Music recommendation and discovery revisited. In *Proceedings of 5th ACM Conference on Recommender Systems*.

Celma, Ò., R. Ramírez, and P. Herrera. 2005. Foafing the music: A music recommendation system based on RSS feeds and user preferences. In *Proceedings of International Society for Music Information Retrieval Conference*, London, UK.

Celma, O., and X. Serra. 2008. FOAFing the music: Bridging the semantic gap in music recommendation. *Web Semantics: Science, Services and Agents on the World Wide Web* 6 (4): 250–6.

Chai, W., and B. Vercoe. 2000. Using user models in music information retrieval systems. In *Proceedings of International Society for Music Information Retrieval Conference*, Plymouth, Massachusetts, USA.

Chang, Y.-I., C.-C. Wu, and M.-C. Tsai. 2011. A user-interests approach to music recommendation. In *Proceedings of the World Congress on Engineering*, Volume 3.

Chao, D. L., J. Balthrop, and S. Forrest. 2005. Adaptive radio: Achieving consensus using negative preferences. In *Proceedings of International ACM SIGGROUP Conference on Supporting Group Work*, 120–3.

Chedrawy, Z., and S. S. R. Abidi. 2006. An adaptive personalized recommendation strategy featuring context sensitive content adaptation. In *Proceedings of 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, 61–70.

Chen, H.-C., and A. L. P. Chen. 2001. A music recommendation system based on music data grouping and user interests. In *Proceedings of 10th International Conference on Information and Knowledge Management*, 231–8.

Chen, H.-C., and A. L. P. Chen. 2005. A music recommendation system based on music and user grouping. *Journal of Intelligent Information Systems* 24 (2-3): 113–32.

Chen, P., C. Tsai, C. Y.N., C. K.C., C. Li, C. Tsai, K. Wu, Y. Chou, C. Li, W. Lin, S. Yu, R. Chiu, C. Lin, C. Wang, P. Wang, W. Su, C. Wu, T. Kuo, T. McKenzie, Y. Chang, C. Ferng, C. Ni, H. Lin, C. Lin, and S. Lin. 2011. A linear ensemble of individual and blended models for music rating prediction. In *KDD-Cup'11 Workshop*.

Chirita, P.-A., W. Nejdl, and C. Zamfir. 2005. Preventing shilling attacks in online recommender systems. In *Proceedings of 7th Annual ACM International Workshop on Web Information and Data Management*, 67–74.

Cho, S.-H., Y.-H. Kim, and J.-B. Park. 2007. Music recommendation system for public places based on sensor network. *International Jounal of Computer Science and Networks Security* 7 (8): 172–80.

Chordia, P., M. Godfrey, and A. Rae. 2008. Extending content-based recommendation: The case of indian classical music. In *Proceedings of International Society for Music Information Retrieval Conference*, 571–6.

Claypool, M., A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. 1999. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR Workshop on Recommender Systems*.

Cohen, W., and W. Fan. 2000. Web-collaborative filtering: Recommending music by crawling the web. *Computer Networks* 33 (1): 685–98.

Crossen, A., J. Budzik, and K. J. Hammond. 2002. Flytrap: Intelligent group music recommendation. In *Proceedings of 7th International Conference on Intelligent User Interfaces*, 184–5.

Cuddy, S., M. Katchabaw, and H. Lutfiyya. 2005. Context-aware service selection based on dynamic and static service attributes. In *Proceedings of IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, Volume 4, 13–20.

Cunningham, S. J., D. Bainbridge, and A. Falconer. 2006. More of an art than a science: Supporting the creation of playlists and mixes. In *Proceedings of International Society for Music Information Retrieval Conference*, Victoria, Canada, 240–5.

Dahlen, B., J. Konstan, J. Herlocker, N. Good, A. Borchers, and J. Riedl. 1998. Jumpstarting movielens: User benefits of starting a collaborative filtering system with "dead data". Technical report, University of Minnesota.

De Lima e Silva, D., M. Medeiros Soares, M. Henriques, M. Schivani Alves, S. de Aguiar, T. de Carvalho, G. Corso, and L. Lucena. 2004. The complex network of the Brazilian popular music. *Physica A: Statistical Mechanics and its Applications* 332: 559–65.

Deerwester, S. C., S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41 (6): 391–407.

Deshpande, M., and G. Karypis. 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems* 22 (1): 143–77.

Dey, A. K. 2001. Understanding and using context. *Personal and Ubiquitous Computing* 5 (1): 4–7.

Dickerson, K., and D. Ventura. 2009. Music recommendation and query-by-content using self-organizing maps. In *Proceedings of International Joint Conference on Neural Networks*, 705–10.

Donaldson, J. 2007a. A hybrid social-acoustic recommendation system for popular music. In *Proceedings of ACM Conference on Recommender Systems*, 187–90.

Donaldson, J. 2007b. Music recommendation mapping and interface based on structural network entropy. In *IEEE 23rd International Conference on Data Engineering Workshop*, 811–7.

Dornbush, S., A. Joshi, Z. Segall, and T. Oates. 2007. A human activity aware learning mobile music player. In *Proceedings of Conference on Advances in Ambient Intelligence*, 107–22.

Dowling, W. 1978. Scale and contour: Two components of a theory of memory for melodies. *Psychological Review* 85 (4): 341–54.

Downie, J. 2003. Music information retrieval. *Annual Review of Information Science and Technology* 37 (1): 295–340.

Dror, G., N. Koenigstein, Y. Koren, and M. Weimer. 2011. The Yahoo! music dataset and KDD-Cup'11. In *Proceedings of KDDCup 2011*.

Dunne, J., L. Lapat, M. Flury, M. Shabib, T. Warner, K. Hammond, and L. Birnbaum. 2002. mpME!: Music recommendation and exploration. In *Proceedings of 7th International Conference on Intelligent User Interfaces*, 235.

Eck, D., P. Lamere, T. Bertin-Mahieux, and S. Green. 2007. Automatic generation of social tags for music recommendation. *Advances in Neural Information Processing Systems* 20: 1–8.

Eggink, J., and G. Brown. 2004. Extracting melody lines from complex audio. In *Proceedings of International Society for Music Information Retrieval Conference*, 84–91.

Elberse, A. 2007. A taste for obscurity: An individual-level examination of "long tail" consumption. Technical report, Harvard Business School.

Fields, B. 2011. *Contextualize Your Listening: The Playlist as Recommendation Engine*. Ph. D. thesis, Goldsmiths University of London, London, UK.

Fields, B., K. Jacobson, C. Rhodes, and M. Casey. 2008. Social playlists and bottleneck measurements : Exploiting musician social graphs using content-based dissimilarity and pairwise maximum flow values. In *Proceedings of International Society for Music Information Retrieval Conference*.

Fields, B., C. Rhodes, and M. d'Inverno. 2010. Using song social tags and topic models to describe and compare playlists. In *Workshop on Music Recommendation and Discovery*.

Firan, C. S., W. Nejdl, and R. Paiu. 2007. The benefit of using tag-based profiles. In *Proceedings of Latin American Web Conference*, 32–41.

Fleder, D. M., and K. Hosanagar. 2007. Recommender systems and their impact on sales diversity. In *Proceedings of 8th ACM Conference on Electronic Commerce*, 192–9.

Flexer, A., D. Schnitzer, M. Gasser, and G. Widmer. 2008. Playlist generation using start and end songs. In *Proceedings of International Society for Music Information Retrieval Conference*, 173–8.

Foote, J. T. 1997. Content-based retrieval of music and audio. *Multimedia Storage and Archiving Systems II, Proceedings of SPIE* 3229: 138–47.

Freed, A. 2006. Music metadata quality: A multiyear case study using the music of skip james. In *Audio Engineering Society Convention 121*.

French, J. C., and D. B. Hauver. 2001. Flycasting: On the fly broadcasting. In *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*.

Fujihara, H., and M. Goto. 2007. A music information retrieval system based on singing voice timbre. In *Proceedings of International Society for Music Information Retrieval Conference*, 467–70.

Gasser, M., and A. Flexer. 2009. Fm4 soundpark audio-based music recommendation in everyday use. In *Proceedings of 6th Sound and Music Computing Conference*, 161–6.

Getoor, L., and M. Sahami. 1999. Using probabilistic relational models for collaborative filtering. In *Proceedings of the Workshop on Web Usage Analysis and User Profiling*.

Gleiser, P., and L. Danon. 2003. Community structure in jazz. *Advances in Complex Systems* 6 (4): 565–73.

Goel, S., A. Broder, E. Gabrilovich, and B. Pang. 2010. Anatomy of the long tail: Ordinary people with extraordinary tastes. In *Proceedings of 3rd ACM International Conference on Web Search and Data Mining*, 201–10.

Goldberg, D., D. Nichols, B. M. Oki, and D. Terry. 1992, December. Using collaborative filtering to weave an information tapestry. *Communications of the ACM* 35 (12): 61–70.

Goldberg, K., T. Roeder, D. Gupta, and C. Perkins. 2001. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval* 4 (2): 133–51.

Goto, M. 2004. A real-time music-scene-description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication* 43 (4): 311–29.

Goto, M., and T. Goto. 2005. Musicream: New music playback interface for streaming, sticking, sorting, and recalling musical pieces. In *Proceedings of International Society for Music Information Retrieval Conference*, 404–11.

Gouyon, F., and S. Dixon. 2005. A review of automatic rhythm description systems. *Computer Music Journal* 29 (1): 34–54.

Green, S. J., P. Lamere, J. Alexander, F. Maillet, S. Kirk, J. Holt, J. Bourque, and X.-W. Mak. 2009. Generating transparent, steerable recommendations from textual descriptions of items. In *Proceedings of 3rd ACM Conference on Recommender Systems*, 281–4.

Guan, D., Q. Li, S. Lee, and Y. Lee. 2006. A context-aware music recommendation agent in smart office. In *Proceedings of 3rd International Conference on Fuzzy Systems and Knowledge Discovery*, Berlin, Heidelberg, 1201–4. Springer-Verlag.

Han, B., S. Rho, S. Jun, and E. Hwang. 2010. Music emotion classification and context-based music recommendation. *Multimedia Tools and Applications* 47 (3): 433–60.

Hayes, C. 2003. *Smart radio: Building community-based Internet music radio*. Ph. D. thesis, Trinity College Dublin, Dublin, Ireland.

Hayes, C., and P. Cunningham. 2000. Smart radio: Building music radio on the fly. In *Expert Systems*, 2–6.

Heckerman, D., D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. 2000. Dependency networks for collaborative filtering and data visualization. In *Proceedings of 16th Conference on Uncertainty in Artificial Intelligence*, 264–73.

Herlocker, J., and J. Konstan. 2001. Content-independent task-focused recommendation. *IEEE Internet Computing* 5 (6): 40–7.

Herlocker, J. L., J. A. Konstan, A. Borchers, and J. Riedl. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of 2nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 230–7.

Herlocker, J. L., J. A. Konstan, and J. Riedl. 2000. Explaining collaborative filtering recommendations. In *Proceedings of ACM Conference on Computer Supported Cooperative Work*, 241–50.

Herlocker, J. L., J. A. Konstan, L. G. Terveen, and J. T. Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transaction on Information Systems* 22 (1): 5–53.

Hijikata, Y., K. Iwahama, and S. Nishida. 2006. Content-based music filtering system with editable user profile. In *Proceedings of ACM Symposium on Applied Computing*, 1050–7.

Hill, W., L. Stead, M. Rosenstein, and G. Furnas. 1995. Recommending and evaluating choices in a virtual community of use. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 194–201.

Hoashi, K., K. Matsumoto, and N. Inoue. 2003. Personalization of user profiles for content-based music retrieval based on relevance feedback. In *Proceedings of 11th ACM International Conference on Multimedia*, 110–9.

Hofmann, T. 1999. Probabilistic latent semantic indexing. In *Proceedings of 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 50–57.

Hofmann, T. 2004. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems* 22 (1): 89–115.

Hu, Y., Y. Koren, and C. Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of 8th IEEE International Conference on Data Mining*, 263–72.

Hu, Y., and M. Ogihara. 2011. Nextone player: A music recommendation system based on user behavior. In *Proceedings of International Society for Music Information Retrieval Conference*, 103–8.

Huang, Y.-C., and S.-K. Jenor. 2004. An audio recommendation system based on audio signature description scheme in MPEG-7 Audio. In *IEEE International Conference on Multimedia and Expo*, Volume 1, 639–42.

Jameson, A., and B. Smyth. 2007. Recommendation to groups. In P. Brusilovsky, A. Kobsa, and W. Nejdl (Eds.), *The Adaptive Web*, Volume 4321 of *Lecture Notes in Computer Science*, 596–627. Springer Berlin Heidelberg.

Jawaheer, G., M. Szomszor, and P. Kostkova. 2010. Comparison of implicit and explicit feedback from an online music recommendation service. In *Proceedings of 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, 47–51.

Kaji, K., K. Hirata, and K. Nagao. 2005. A music recommendation system based on annotations about listeners' preferences and situations. In *Proceedings of 1st International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, 231–4.

Kaminskas, M., and F. Ricci. 2011. Location-adapted music recommendation using tags. In *User Modeling, Adaption and Personalization*, Volume 6787, 183–94.

Karypis, G. 2001. Evaluation of item-based top-n recommendation algorithms. In *Proceedings of 10th International Conference on Information and Knowledge Management*, 247–54.

Khine, S. Z. K., T. L. Nwe, and H. Li. 2008. Exploring perceptual based timbre feature for singer identification. In *Computer Music Modeling and Retrieval. Sense of Sounds*, Volume 4969, 159–71.

Kim, D.-M., K.-S. Kim, K.-H. Park, J.-H. Lee, and K.-M. Lee. 2007. A music recommendation system with a dynamic k-means clustering algorithm. In *Proceedings of 6th International Conference on Machine Learning and Applications*, 399–403.

Kim, J.-H., K.-Y. Chung, J.-K. Ryu, K.-W. Rim, and J.-H. Lee. 2008. Personal history based recommendation service system with collaborative filtering. In *Proceedings of Advanced Software Engineering and Its Applications*, 261 –4.

Kim, J.-H., K.-Y. Jung, and J.-H. Lee. 2006. Hybrid music filtering for recommendation based ubiquitous computing environment. In *Proceedings of 5th International Conference on Rough Sets and Current Trends in Computing*, Volume 4259, 796–805.

Kim, J.-H., U.-G. Kang, and J.-H. Lee. 2007. Content-based filtering for music recommendation based on ubiquitous computing. In *Proceedings of International Federation for Information Processing*, Volume 228, 463–72.

Kim, J.-H., C.-W. Song, K.-W. Lim, and J.-H. Lee. 2006. Design of music recommendation system using context information. In *Proceedings of 9th Pacific Rim International Conference on Agent Computing and Multi-Agent Systems*, Volume 4088, 708–13.

Kim, J.-K., H.-K. Kim, H.-Y. Oh, and Y.-U. Ryu. 2010. A group recommendation system for online communities. *International Journal of Information Management* 30 (3): 212–9.

Kim, K., D. Lee, T.-B. Yoon, and J.-H. Lee. 2008. A music recommendation system based on personal preference analysis. In *Proceedings of 1st International Conference on Applications of Digital Information and Web Technologies*, 102–6.

Knees, P., T. Pohle, M. Schedl, and G. Widmer. 2006. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. In *Proceedings of 8th ACM International Workshop on Multimedia Information Retrieval*, 147–54.

Kodama, Y., S. Gayama, Y. Suzuki, S. Odagawa, T. Shioda, F. Matsushita, and T. Tabata. 2005. A music recommendation system. In *International Conference on Consumer Electronics*, 219–20.

Koenigstein, N., G. Dror, and Y. Koren. 2011. Yahoo! music recommendations: Modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of 5th ACM Conference on Recommender Systems*, 165–72.

Kohavi, R., R. Longbotham, D. Sommerfield, and R. Henne. 2009. Controlled experiments on the web: Survey and practical guide. *Data Mining and Knowledge Discovery* 18 (1): 140–81.

Kuo, F.-F., M.-F. Chiang, M.-K. Shan, and S.-Y. Lee. 2005. Emotion-based music recommendation by association discovery from film music. In *Proceedings of 13th Annual ACM International Conference on Multimedia*, 507–10.

Kuo, F.-F., and M.-K. Shan. 2002. A personalized music filtering system based on melody style classification. In *IEEE International Conference on Data Mining*, 649–52.

Lam, S. K., D. Frankowski, and J. Riedl. 2006. Do you trust your recommendations? an exploration of security and privacy issues in recommender systems. In *Emerging Trends in Information and Communication Security*, Volume 3995 of *Lecture Notes in Computer Science*, 14–29. Springer.

Lam, S. K., and J. Riedl. 2004. Shilling recommender systems for fun and profit. In *Proceedings of 13th International Conference on World Wide Web*, 393–402.

Lambiotte, R., and M. Ausloos. 2006. On the genre-fication of music: A percolation approach. *The European Physical Journal B - Condensed Matter and Complex Systems* 50 (1): 183–8.

Lamere, P. 2008. Social tagging and music information retrieval. *Journal of New Music Research* 37 (2): 101–14.

Lee, D., S. Park, M. Kahng, S. Lee, and S. Lee. 2010. Exploiting contextual information from event logs for personalized recommendation. In *Computer and Information Science 2010*, Volume 317, 121–39.

Lee, J. H. 2011. How similar is too similar?: Exploring users' perceptions of similarity in playlist evaluation. In *Proceedings of International Society for Music Information Retrieval Conference*, 109–14.

Lee, J. S., and J. C. Lee. 2006. Music for my mood: A music recommendation system based on context reasoning. *Smart Sensing and Context* 4272: 190–203.

Lee, J. S., and J. C. Lee. 2007. Context awareness by case-based reasoning in a music recommendation system. In *Proceedings of 4th International Conference on Ubiquitous Computing Systems*, Volume 4836, 45–58.

Lehtiniemi, A. 2008. Evaluating SuperMusic: Streaming context-aware mobile music service. In *Proceedings of International Conference on Advances in Computer Entertainment Technology*, 314–21.

Lemire, D., and A. Maclachlan. 2005. Slope one predictors for online rating-based collaborative filtering. In *Proceedings of the SIAM Data Mining*.

Lesaffre, M., M. Leman, and J.-P. Martens. 2006. A user-oriented approach to music information retrieval. In *Dagstuhl Seminar Proceedings*, 1–11.

Levy, M., and K. Bosteels. 2010. Music recommendation and the long tail. In *Proceedings of Workshop on Music Recommendation and Discovery*.

Levy, M., and M. Sandler. 2008. Learning latent semantic models for music from social tags. *Journal of New Music Research* 37 (2): 137–50.

Levy, M., and M. Sandler. 2009. Music information retrieval using social tags and audio. *IEEE Transactions on Multimedia* 11 (3): 383–95.

Li, Q., B. M. Kim, D. H. Guan, and D. W. Oh. 2004. A music recommender based on audio features. In *Proceedings of 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 532–3.

Li, Q., S.-H. Myaeng, D. H. Guan, and B. M. Kim. 2005. A probabilistic model for music recommendation considering audio features. In *Proceedings of 2nd Asia Conference on Asia Information Retrieval Technology*, Volume 3689, Berlin, Heidelberg, 72–83. Springer-Verlag.

Li, Q., S.-H. Myaeng, and B. M. Kim. 2007. A probabilistic music recommender considering user opinions and audio features. *Information Processing & Management* 43 (2): 473–87.

Li, T., and M. Ogihara. 2005. Music genre classification with taxonomy. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Volume 5, 197–200.

Linden, G., B. Smith, and J. York. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE* 7 (1): 76–80.

Liu, H., J. Hu, and M. Rauterberg. 2009. Music playlist recommendation based on user heartbeat and music preference. In *Proceedings of International Conference on Computer Technology and Development*, Volume 1, 545–49.

Liu, N.-H., S.-W. Lai, C.-Y. Chen, and S.-J. Hsieh. 2009. Adaptive music recommendation based on user behavior in time slot. *International Journal of Computer Science and Networks Security* 9 (2).

Loeb, S. 1992. Architecting personalized delivery of multimedia information. *Communications of the ACM* 35 (12): 39–47.

Logan, B. 2002. Content-based playlist generation: Exploratory experiments. In *Proceedings of International Society for Music Information Retrieval Conference*, Paris, France.

Logan, B. 2004. Music Recommendation from Song Sets. In *Proceedings of International Society for Music Information Retrieval Conference*.

Logan, B., and A. Salomon. 2001. A music similarity function based on signal analysis. In *International Conference on Multimedia and Expo*, 745–8.

Lu, C.-C., and V. S. Tseng. 2009. A novel method for personalized music recommendation. *Expert Systems with Applications* 36 (6): 10035–44.

Magno, T., and C. Sable. 2008. A comparison of signal based music recommendation to genre labels, collaborative filtering, musicological analysis, human recommendation and random baseline. In *Proceedings of International Society for Music Information Retrieval Conference*, 161–6.

Maillet, F., D. Eck, G. Desjardins, and P. Lamere. 2009. Steerable playlist generation by learning song similarity from radio station playlists. In *Proceedings of International Society for Music Information Retrieval Conference*, 345–50.

McCarthy, J. F., and T. D. Anagnost. 1998. Musicfx: An arbiter of group preferences for computer supported collaborative workouts. In *Proceedings of ACM Conference on Computer Supported Cooperative Work*, 363–72.

McEnnis, D., and S. J. Cunningham. 2007. Sociology and music recommendation systems. In *Proceedings of International Society for Music Information Retrieval Conference*, 185–6.

McFee, B., T. Bertin-Mahieux, D. P. Ellis, and G. R. Lanckriet. 2012. The million song dataset challenge. In *Proceedings of 21st International Conference Companion on World Wide Web*, 909–16.

McFee, B., and G. R. G. Lanckriet. 2009. Heterogeneous embedding for subjective artist similarity. In *Proceedings of International Society for Music Information Retrieval Conference*, 513–8.

McFee, B., and G. R. G. Lanckriet. 2011. The natural language of playlists. In *Proceedings of International Society for Music Information Retrieval Conference*, 537–42.

McKay, C., and I. Fujinaga. 2006. Musical genre classification: Is it worth pursuing and how can it be improved. In *Proceedings of International Society for Music Information Retrieval Conference*, 101–107.

McKenzie, T., C. Ferng, Y. Chen, C. Li, C. Tsai, K. Wu, Y. Chang, C. Li, W. Lin, and S. Yu. 2011. Novel models and ensemble techniques to discriminate favorite items from unrated ones for personalized music recommendation.

McNee, S. M., J. Riedl, and J. A. Konstan. 2006. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *CHI Extended Abstracts on Human Factors in Computing Systems*, 1097–101.

Melville, P., R. J. Mooney, and R. Nagarajan. 2002. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of 18th National Conference on Artificial Intelligence*, 187–92.

Melville, P., and V. Sindhwani. 2010. Recommender systems. In C. Sammut and G. I. Webb (Eds.), *Encyclopedia of Machine Learning*, 829–38. Berlin Heidelberg: Springer-Verlag.

Milgram, S. 1967. The small world problem. *Psychology today* 2 (1): 60–7.

Milicevic, A., A. Nanopoulos, and M. Ivanovic. 2010. Social tagging in recommender systems: A survey of the state-of-the-art and possible extensions. *Artificial Intelligence Review* 33 (3): 187–209.

Mims, C. 2010. How iTunes Genius really works. `http://www.technologyreview.com/view/419198/how-itunes-genius-really-works`. Last accessed 10 Oct 2012.

Mitrović, D., M. Zeppelzauer, and C. Breiteneder. 2010. Features for content-based audio retrieval. *Advances in Computers* 78: 71–150.

Mobasher, B., R. Burke, R. Bhaumik, and C. Williams. 2005. Effective attack models for shilling item-based collaborative filtering systems. In *Proceedings of the Workshop on Knowledge Discovery in the Web*.

Morchen, F., A. Ultsch, M. Thies, and I. Lohken. 2006. Modeling timbre distance with temporal statistics from polyphonic music. *IEEE Transactions on Audio, Speech, & Language Processing* 14 (1): 81 – 90.

Nanopoulos, A., D. Rafailidis, P. Symeonidis, and Y. Manolopoulos. 2010. Musicbox: Personalized music recommendation based on cubic analysis of social tags. *IEEE Transactions on Audio, Speech, & Language Processing* 18 (2): 407–12.

Ness, S. R., A. Theocharis, G. Tzanetakis, and L. G. Martins. 2009. Improving automatic music tag annotation using stacked generalization of probabilistic SVM outputs. In *Proceedings of 17th ACM International Conference on Multimedia*, 705–8.

Niitsuma, M., H. Takaesu, H. Demachi, M. Oono, and H. Saito. 2008. Development of an automatic music selection system based on runner's step frequency. In *Proceedings of International Society for Music Information Retrieval Conference*, 193–98.

North, A., and D. Hargreaves. 1996. Situational influences on reported musical preference. *Psychomusicology: Music, Mind and Brain* 15 (1-2): 30–45.

North, A., D. Hargreaves, and J. Hargreaves. 2004. Uses of music in everyday life. *Music Perception* 22 (1): 41–77.

O'Connor, M., D. Cosley, J. A. Konstan, and J. Riedl. 2002. Polylens: A recommender system for groups of users. In W. Prinz, M. Jarke, Y. Rogers, K. Schmidt, and V. Wulf (Eds.), *Proceedings of 7th European Conference on Computer-Supported Cooperative Work*, 199–218. Springer Netherlands.

Oliver, N., and L. Kreger-Stickles. 2006. PAPA: Physiology and purpose-aware automatic playlist generation. In *Proceedings of International Society for Music Information Retrieval Conference*, 250–3.

Orio, N. 2006. Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval* 1 (1): 1–96.

Pachet, F. 2003. Content management for electronic music distribution. *Communications of the ACM* 46 (4): 71–5.

Pachet, F. 2005. Knowledge management and musical metadata. In D. Schwartz (Ed.), *Encyclopedia of Knowledge Management*. Hershey, PA: Idea Group.

Pachet, F., P. Roy, and D. Cazaly. 2000. A combinatorial approach to content-based music selection. *IEEE Multimedia* 7 (1): 44–51.

Pampalk, E. 2001. Islands of music: Analysis, organization, and visualization of music archives. Master's thesis, Vienna University of Technology, Vienna, Austria.

Pampalk, E., A. Flexer, and G. Widmer. 2005. Improvements of audio-based music similarity and genre classification. In *Proceedings of International Society for Music Information Retrieval Conference*.

Pampalk, E., and M. Gasser. 2006. An implementation of a simple playlist generator based on audio similarity measures and user feedback. In *Proceedings of International Society for Music Information Retrieval Conference*, Victoria, Canada, 389–90.

Pampalk, E., and M. Goto. 2006. Musicrainbow: A new user interface to discover artists using audio-based similarity and web-based labeling. In *Proceedings of International Society for Music Information Retrieval Conference*, 367–70.

Pampalk, E., and M. Goto. 2007. MusicSun: A new approach to artist recommendation. In *Proceedings of International Society for Music Information Retrieval Conference*, Vienna, Austria.

Pampalk, E., T. Pohle, and G. Widmer. 2005. Dynamic playlist generation based on skipping behavior. In *Proceedings of International Society for Music Information Retrieval Conference*, London, UK.

Park, C. H., and M. Kahng. 2010. Temporal dynamics in music listening behavior: A case study of online music service. In *Proceedings of IEEE/ACIS 9th International Conference on Computer and Information Science*, 573–8.

Park, J., O. Celma, M. Koppenberger, P. Cano, and J. M. Buldu. 2007. The social network of contemporary popular musicians. *International Journal of Bifurcation and Chaos* 17: 2281–8.

Park, M. K. S., and N. Moon. 2011. The effects of personal sentiments and contexts on the acceptance of music recommender systems. In *Proceedings of 5th FTRA International Conference on Multimedia and Ubiquitous Engineering*, 289–92.

Paulus, J., M. Mller, and A. Klapuri. 2010. State of the art report: Audio-based music structure analysis. In *Proceedings of International Society for Music Information Retrieval Conference*, 625–36.

Pauws, S., and B. Eggen. 2003. Realization and user evaluation of an automatic playlist generator. *Journal of New Music Research* 32 (2): 179–92.

Pauws, S., and S. van de Wijdeven. 2005. User evaluation of a new interactive playlist generation concept. In *Proceedings of International Society for Music Information Retrieval Conference*, London, UK, 638–43.

Pauws, S., W. Verhaegh, and M. Vossen. 2006. Fast generation of optimal music playlists using local search. In *Proceedings of International Society for Music Information Retrieval Conference*, Victoria, Canada, 138–43.

Pavlov, D., and D. M. Pennock. 2002. A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In *Proceedings of 16th Annual Conference on Neural Information Processing Systems*, 1441–8.

Pennock, D. M., E. Horvitz, S. Lawrence, and C. L. Giles. 2000. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of 16th Conference on Uncertainty in Artificial Intelligence*.

Perik, E., B. de Ruyter, P. Markopoulos, and B. Eggen. 2004. The sensitivities of user profile information in music recommender systems. In *Proceedings of Private, Security, Trust*, 137–41.

Pestoni, F., J. Wolf, M. Habib, and A. Mueller. 2001. KARC: Radio research. In *Proceedings of 1st International Conference on Web Delivering of Music*, 139–46.

Platt, J. C., C. J. C. Burges, S. Swenson, C. Weare, and A. Zheng. 2002. Learning a gaussian process prior for automatically generating music playlists. In *Proceedings of Advances in Neural Information Processing Systems*, Volume 14, 1425–32.

Poliner, G., and D. Ellis. 2005. A classification approach to melody transcription. In *Proceedings of International Society for Music Information Retrieval Conference*, 161–6.

Popescul, A., D. M. Pennock, and S. Lawrence. 2001. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of 17th Conference on Uncertainty in Artificial Intelligence*, 437–44.

Ragno, R., C. Burges, and C. Herley. 2005. Inferring similarity between music objects with application to playlist generation. In *Proceedings of 7th ACM SIGMM International Workshop on Multimedia Information Retrieval*, 73–80.

Rashid, A. M., S. K. Lam, G. Karypis, and J. Riedl. 2006. ClustKNN: A highly scalable hybrid model- & memory-based cf algorithm. In *Proceedings of 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Resnick, P., N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. 1994. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM Conference on Computer Supported Cooperative Work*, 175–86.

Resnick, P., and H. R. Varian. 1997. Recommender systems. *Communications of the ACM* 40 (3): 56–8.

Rho, S., S. Song, E. Hwang, and M. Kim. 2009. COMUS: Ontological and rule-based reasoning for music recommendation system. In *Advances in Knowledge Discovery and Data Mining*, Volume 5476, 859–66.

Sandvold, V., T. Aussenac, Ò. Celma, and P. Herrera. 2006. Good vibrations: Music discovery through personal musical concepts. In *Proceedings of International Society for Music Information Retrieval Conference*, Victoria, Canada.

Sarwar, B., G. Karypis, J. Konstan, and J. Reidl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of 10th International Conference on World Wide Web*, 285–95.

Sarwar, B., G. Karypis, J. Konstan, and J. Riedl. 2000. Analysis of recommendation algorithms for e-commerce. In *Proceedings of 2nd ACM Conference on Electronic Commerce*, 158–67.

Schein, A. I., A. Popescul, L. H. Ungar, and D. M. Pennock. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 253–60.

Scheirer, E., and M. Slaney. 1997. Construction and evaluation of a robust multifeature speech/music discriminator. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Volume 2, 1331–4.

Schwarz, D., and X. Rodet. 1999. Spectral envelope estimation and representation for sound analysis-synthesis. In *Proceedings of International Computer Music Conference*.

Seyerlehner, K., P. Knees, D. Schnitzer, and G. Widmer. 2009. Browsing music recommendation networks. In *Proceedings of International Society for Music Information Retrieval Conference*.

Shan, M.-K., F.-F. Kuo, M.-F. Chiang, and S.-Y. Lee. 2009. Emotion-based music recommendation by affinity discovery from film music. *Expert Systems with Applications* 36 (4): 7666–74.

Shani, G., R. I. Brafman, and D. Heckerman. 2002. An MDP-based recommender system. In *Proceedings of 18th Conference on Uncertainty in Artificial Intelligence*, 453–60.

Shani, G., and A. Gunawardana. 2011. Evaluating recommendation systems. *Recommender Systems Handbook*: 257–97.

Shao, B., M. Ogihara, D. Wang, and T. Li. 2009. Music recommendation based on acoustic features and user access patterns. *IEEE Transactions on Audio, Speech & Language Processing* 17 (8): 1602–11.

Shardanand, U. 1994. Social information filtering for music recommendation. Master's thesis, Massachusetts Institute of Technology.

Shardanand, U., and P. Maes. 1995. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 210–7.

Sheth, B., and P. Maes. 1993. Evolving agents for personalized information filtering. In *Proceedings of 9th IEEE Conference on Artificial Intelligence for Applications*.

Shin, D., J. Lee, J. Yeon, and S. Lee. 2009. Context-aware recommendation by aggregating user context. In *Proceedings of IEEE Conference on Commerce and Enterprise Computing*, 423–30.

Sinha, R., and K. Swearingen. 2002. The role of transparency in recommender systems. In *CHI Extended Abstracts on Human Factors in Computing Systems*, 830–1.

Slaney, M., and W. White. 2006. Measuring playlist diversity for recommendation systems. In *Proceedings of 1st ACM Workshop on Audio and Music Computing Multimedia*, 77–82.

Smyth, B., and P. McClave. 2001. Similarity vs. diversity. In D. Aha and I. Watson (Eds.), *Case-Based Reasoning Research and Development*, Volume 2080, 347–61.

Soundscan, N. 2007. State of the industry. *Nielsen Soundscan Report. National Association of Recording Merchandisers*.

Su, J.-H., H.-H. Yeh, P. S. Yu, and V. S. Tseng. 2010. Music recommendation using content and context information mining. *IEEE Intelligent Systems* 25 (1): 16–26.

Su, X., and T. Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* 2009.

Swearingen, K., and R. Sinha. 2001. Beyond algorithms: An HCI perspective on recommender systems. In *ACM SIGIR Workshop on Recommender Systems*, Volume 13, 393–408.

Symeonidis, P., M. M. Ruxanda, A. Nanopoulos, and Y. Manolopoulos. 2008. Ternary semantic analysis of social tags for personalized music recommendation. In *Proceedings of International Society for Music Information Retrieval Conference*, 219–24.

Tan, S., J. Bu, C. Chen, B. Xu, C. Wang, and X. He. 2011. Using rich social media information for music recommendation via hypergraph model. *ACM Transactions on Multimedia Computing, Communications, and Applications* 7 (1): 22.

Tiemann, M., S. Pauws, and F. Vignoli. 2007. Ensemble learning for hybrid music recommendation. In S. Dixon, D. Bainbridge, and R. Typke (Eds.), *Proceedings of International Society for Music Information Retrieval Conference*, 179–80.

Torrens, M., P. Hertzog, and J. L. Arcos. 2004. Visualizing and exploring personal music libraries. In *Proceedings of International Society for Music Information Retrieval Conference*.

Tsinaraki, C., and S. Christodoulakis. 2005. Semantic user preference descriptions in MPEG-7/21. In *Hellenic Data Management Symposium*.

Turnbull, D., L. Barrington, and G. R. G. Lanckriet. 2008. Five approaches to collecting tags for music. In *Proceedings of International Society for Music Information Retrieval Conference*, 225–30.

Tzanetakis, G., and P. Cook. 2002. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing* 10 (5): 293–302.

Uitdenbogerd, A., and R. van Schnydel. 2002. A review of factors affecting music recommender success. In *Proceedings of International Society for Music Information Retrieval Conference*, Paris, France.

Ungar, L. H., and D. P. Foster. 1998. Clustering methods for collaborative filtering. In *Workshop on Recommender Systems at the 15th National Conference on Artificial Intelligence*, 112–25.

van Gulik, R., and F. Vignoli. 2005. Visual playlist generation on the artist map. In *Proceedings of International Society for Music Information Retrieval Conference*, London, UK, 520–3.

van Gulik, R., F. Vignoli, and H. van de Wetering. 2004. Mapping music in the palm of your hand, explore and discover your collection. In *Proceedings of International Society for Music Information Retrieval Conference*.

Vassileva, J. 1994. A practical architecture for user modeling in a hypermedia-based information system. In *Proceedings of 4th International Conference on User Modeling*, 115–20.

Vembu, S., and S. Baumann. 2005. A self-organizing map based knowledge discovery for music recommendation systems. In *Proceedings of 2nd International Conference on Computer Music Modeling and Retrieval*, 119–29.

Vignoli, F., and S. Pauws. 2005. A music retrieval system based on user driven similarity and its evaluation. In *Proceedings of International Society for Music Information Retrieval Conference*, London, UK, 272–9.

Vucetic, S., and Z. Obradovic. 2005. Collaborative filtering using a regression-based approach. *Knowledge & Information Systems* 7 (1): 1–22.

Whitman, B., and S. Lawrence. 2002. Inferring descriptions and similarity for music from community metadata. In *Proceedings of International Computer Music Conference*.

Wijnalda, G., S. Pauws, F. Vignoli, and H. Stuckenschmidt. 2005. A personalized music system for motivation in sport performance. *IEEE Pervasive Computing* 4 (3): 26–32.

Winckel, F., and T. Binkley. 1967. *Music, sound and sensation: A modern exposition*. Dover Publications.

Xu, C., N. Maddage, and X. Shao. 2005. Automatic music classification and summarization. *IEEE Transactions on Speech and Audio Processing* 13 (3): 441–50.

Yang, Y., and J. Li. 2005. Interest-based recommendation in digital library. *Journal of Computer Science* 1 (1): 40–6.

Yapriady, B., and A. Uitdenbogerd. 2005. Combining demographic data with collaborative filtering for automatic music recommendation. *Knowledge-Based Intelligent Information and Engineering Systems* 3684: 201–7.

Yoon, T., S. Lee, K. ho Yoon, D. Kim, and J.-H. Lee. 2008. A personalized music recommendation system with a time-weighted clustering. In *Proceedings of 4th International IEEE Conference on Intelligent Systems*, Volume 2, 48–52.

Yoshii, K., and M. Goto. 2009. Continuous pLSI and smoothing techniques for hybrid music recommendation. In *Proceedings of International Society for Music Information Retrieval Conference*, 339–44.

Yoshii, K., M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. 2006. Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. In *Proceedings of International Society for Music Information Retrieval Conference*, Victoria, Canada, 296–301.

Yoshii, K., M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. 2007. Improving efficiency and scalability of model-based music recommender system based on incremental training. In *Proceedings of International Society for Music Information Retrieval Conference*, 89–94.

Yoshii, K., M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. 2008. An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model. *IEEE Transactions on Audio, Speech & Language Processing* 16 (2): 435–47.

Yu, Z., X. Zhou, and D. Zhang. 2005. An adaptive in-vehicle multimedia recommender for group users. In *Proceedings of IEEE 61st Vehicular Technology Conference*, Volume 5, 2800–4.

Zanin, M., P. Cano, J. M. Buldú, and O. Celma. 2008. Complex networks in recommendation systems. In *Proceedings of 2nd WSEAS International Conference on Computer Engineering and Applications*, 120–24.

Zhang, M., and N. Hurley. 2008. Avoiding monotony: Improving the diversity of recommendation lists. In *Proceedings of ACM Conference on Recommender Systems*, 123–30.

Zhang, Y. C., D. O. Séaghdha, D. Quercia, and T. Jambor. 2012. Auralist: Introducing serendipity into music recommendation. In *Proceedings of 5th ACM International Conference on Web Search and Data Mining*, 13–22.

Zhu, X., Y. Y. Shi, H. G. Kim, and K. W. Eom. 2006. An integrated music recommendation system. *IEEE Transactions on Consumer Electronics* 52 (3): 917–25.

Ziegler, C.-N., S. M. McNee, J. A. Konstan, and G. Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of 14th International Conference on World Wide Web*, 22–32.