Available online at www.sciencedirect.com
ScienceDirect
Information Processing and Management 43 (2007) 473–487
INFORMATION PROCESSING & MANAGEMENT
www.elsevier.com/locate/infoproman

A probabilistic music recommender considering user opinions and audio features

Qing Li [a,*], Sung Hyon Myaeng [a], Byeong Man Kim [b]

[a] *Information and Communications University, Daejeon, 305-732, Republic of Korea*
[b] *Kumoh National Institute of Technology, Kumi, 730-701, Republic of Korea*

Received 27 May 2006; accepted 25 July 2006
Available online 9 October 2006

Abstract

A recommender system has an obvious appeal in an environment where the amount of on-line information vastly outstrips any individual's capability to survey. Music recommendation is considered a popular application area. In order to make personalized recommendations, many collaborative music recommender systems (CMRS) focus on capturing precise similarities among users or items based on user historical ratings. Despite the valuable information from audio features of music itself, however, few studies have investigated how to utilize information extracted directly from music for personalized recommendation in CMRS. In this paper, we describe a CMRS based on our proposed item-based probabilistic model, where items are classified into groups and predictions are made for users considering the Gaussian distribution of user ratings. In addition, this model has been extended for improved recommendation performance by utilizing audio features that help alleviate three well-known problems associated with data sparseness in collaborative recommender systems: user bias, non-association, and cold start problems in capturing accurate similarities among items. Experimental results based on two real-world data sets lead us to believe that content information is crucial in achieving better personalized recommendation beyond user ratings. We further show how primitive audio features can be combined into aggregate features for the proposed CRMS and analyze their influences on recommendation performance. Although this model was developed originally for music collaborative recommendation based on audio features, our experiment with the movie data set demonstrates that it can be applied to other domains.

Keywords: Collaborative filtering; Music recommender system; Probabilistic model; Information filtering

1. Introduction

With the development of e-commerce and the proliferation of easily accessible information, recommender systems have become a popular technique to prune large information spaces so that users are directed toward

* Corresponding author.
E-mail address: liqing@icu.ac.kr (Q. Li).

0306-4573/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2006.07.005

those items that best meet their needs and preferences. The technique has been an integral part of e-commerce sites such as Amazon, Yahoo and CDNow.

At the initial stage, recommender systems mostly relied on a content-based filtering (CBF) mechanism. It selects the right information for a user by matching the user preference, which may be implicit or explicit, against databases. For example, a search engine recommends web pages whose contents are similar to a user query (Wittenburg, Das, Hill, & Stead, 1995). Despite the efficiency of a CBF in locating textual items relevant to a topic, it is a little difficult for such a system to be directly applied to multimedia when they lack textual descriptions such as a music genre, title, composer, or singer. In addition, representing a user preference or a query for non-textual features is also a big challenge for a music recommender system (MRS). Some researchers (Ghias, Logan, Chamberlin, & Smith, 1995) suggested a method of querying an audio database by humming. Not all users, however, have a gift of humming the melody of their favorite songs for searching.

Collaborative filtering (CF), as in GroupLens (Resnick, Iacovou, Suchak, Bergstorm, & Riedl, 1994) and Ringo (Shardanand & Maes, 1995), has been considered a mainstream technique for recommender systems for a long time until now. CF uses opinions of others to predict the interest of a user. A target user is matched against the database of user histories to discover other users, called neighbors, who have historically similar tastes. Items the neighbors like are then recommended to the target user. For instance, the GAB system (Wittenburg et al., 1995) recommends web pages based on the bookmarks; the Jeter system recommends jokes (Gupta, Dhruv, Mark Digiovanni, Hiro Narita, & Jester, 1999); MovieLens recommends movies (http://movielens.umn.edu); and Flycasting recommends online radio programs (Hauver, 2001). Most of prevalent CF systems focus on calculating the user–user similarity to make predictions, which is the so-called user-based CF. However, Sarwar (Sarwar, Karypis, Konstan, & Riedl, 2001) has proven that an item-based CF is better than a user-based CF on precision and computational complexity.

Despite the popularity of CF techniques, researchers also realized that the content information of items did help providing a good recommendation service. The idea of a hybrid system capitalizing on both CF and CBF has been suggested. Examples include Fab (Balabanovic & Shoham, 1997), ProfBuilder (Wasfi, 1999), RAAP (Delgado, Ishii, & Ura, 1998), and Chen's music recommender (Chen & Hung-Chen, 2001).

The Fab system (Balabanovic & Shoham, 1997) uses content-based techniques instead of user ratings to create user profiles. Since the quality of predictions is fully dependent on the content, inaccurate profiles result in inaccurate correlations with other users, yielding poor predictions. ProfBuilder (Wasfi, 1999) recommends web pages using both content-based and collaborative filters, each of which creates a recommendation list separately to make a combined prediction. Claypool et al. (1999) apply a hybrid approach for an online newspaper domain, combining two types of predictions using an adaptive weighted average. As the number of users accessing an item increases, the weight of the collaborative component tends to increase. However, the authors do not clearly describe how to decide the weights of collaborative and content-based components. RAAP (Delgado et al., 1998) is a content-based collaborative information filtering system that helps the user classify domain specific information found on the WWW, and also recommends those URLs to other users with similar interests. To determine the similarity of interests among users, a scalable Pearson correlation algorithm based on the web page category is used.

In Chen's music recommender system (Chen & Hung-Chen, 2001), both user interests and music features are applied for making recommendations. Access histories of users are analyzed to mine user interests. This work makes a content-based and collaborative recommendation, emphasizing on the degree to which a user favors a music group. This approach assumes that each piece of music is assigned to a group. If a user is interested in a certain group, the songs in the group are recommended to the user with an equal value. In other words, this system can only recommend an un-ordered list of music pieces to users.

Clustering is the key idea of CF including hybrid systems that rely on CF. Clustering is implemented in various ways, either explicitly or implicitly in most CF techniques. A pioneering CF method (Resnick et al., 1994) based on Pearson correlation coefficient aims at grouping the users with similar tastes for an active user. It first calculates the similarities between the active user and others, and then clusters users into groups with similar tastes. There are several variants (Breese, Heckerman, & Kardie, 1998; Li, Qing, Kim, Guan, & Oh, 2004; Sarwar et al., 2001) derived from the CF method. Since relationships should be calculated and saved in the memory for recommendation, these methods are called as memory-based techniques.

Probabilistic models exploit explicit clustering to model the preferences of the underlying users, from which predictions are inferred. Examples include the Bayesian clustering model (Breese et al., 1998), the Bayesian network model (Breese et al., 1998), and the Aspect model (Hofmann, 2003). The basic assumption of the Bayesian clustering model is that ratings are observations of the multinomial mixture model with parameters, and the model is estimated by using EM. The Bayesian network model aims at capturing item dependency. Each item is a node and dependency structure is learned from observables. The aspect model is based on a latent cause model that introduces the notion of user communities or groups of items.

Besides the above methods where clustering is integrated into some probabilistic or memory-based techniques, some researchers also apply a data clustering algorithm to the process of rating data in CF as an isolated step. O'Conner and Herlocker (1999) uses existing data clustering algorithms to do item clustering based on user ratings, and then predictions are calculated independently within each partition. SWAMI (Fisher et al., 2000) applies the meta-user mechanism for recommendation by first grouping users and creating a profile for each cluster as a meta-user. Predictions are calculated with a Pearson correlation method where only meta-users are considered as potential neighbors. Li and Kim (2003) and Li and Kim (2004) have proposed to apply a clustering method to include content information for CF. This approach groups users based on the content information extracted from user profiles and treats the membership between users and these groups as the pseudo-ratings for final recommendation.

While the aforementioned systems and methods cluster users or items, most of them fail to handle the well-known problems in CF: non-association, user bias, and cold start. Since these three problems occur in both item-based and user-based CF, we address them in the context of item-based CF only.

The first challenge is to deal with non-association in the data. For instance, if two similar items have never been wanted by the same user, their relationship is not known explicitly. In pure item-based CF, 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.

The second one is to handle user biases in the past ratings. For example, as shown in Table 1, Music 3 and Music 4 have the same history in ratings and thus have the same opportunity to be recommended to the user Jack by an item-based CF system. If the system knows that Music 3 falls into the Rock category together with Music 1 and Music 2, however, it can recommend Music 3 to Jack with a higher priority, provided that he is known to like Rock music in the past as in the example. User ratings in the history are not just comprehensive enough.

The third one is the cold start problem. It is hard for pure CF to recommend a new item that has no history, i.e., no user opinions. Music 5 in Table 1 is an example.

In this paper, we describe a collaborative music recommender system (CMRS) based on our previously proposed item-based probabilistic model (Li & Kim, 2005; Li, Kim, Kim, & Kim, 2004). It has been extended for improved recommendation performance by utilizing audio features that help alleviate the aforementioned three well-known problems associated with data sparseness in collaborative recommender systems: user bias, non-association, and cold start problems in capturing accurate similarities among items. Experimental results based on two real-world data sets lead us to believe that content information is crucial in achieving better personalized recommendation beyond user ratings. We further show how primitive audio features can be combined into aggregate features for the proposed CRMS and analyze their influences on recommendation performance.

Table 1
Rating information

Item ID	Jack	Oliver	Peter	Tom	Rock	Country
Music 1	5	4	3		Y	N
Music 2	4	4	3		Y	N
Music 3		4	3		Y	N
Music 4		4	3		N	Y
Music 5					Y	N
Music 6				4	Y	N

2. Probabilistic model

The intuition of item-based collaborative filtering is that since most of those users who enjoyed "Star Wars" would also be fond of a movie like "Independence Day", and you liked "Star Wars", you may also enjoy "Independency Day". Therefore, a recommender system should group similar items into an item community for which most users share similar feelings. If we can successfully cluster items into proper item communities, recommendations could be easily made.

However, it is not easy to cluster items properly due to the sparseness of user ratings, which is also the cause for the three problems mentioned in the previous section. They occur quite often and negatively affect the clustering result and hence recommendation. As shown in Table 1, the attributes of items such as their genre would help us alleviate those problems. As an example for non-association, the relationships between Music 6 and others are unknown. However, with the help of the music genre information, we can partially infer that Music 6 has a close relationship with other music pieces except Music 4. The other two problems can be similarly handled by virtue of the genre information. It would be helpful to consider the item attributes for clustering instead of user ratings alone. Therefore, we developed a probabilistic model (Li & Kim, 2005) that takes into account the grouping idea based on ratings and item attributes together.

In this model, items are classified into groups based on both content information and ratings together, and predictions are made considering the Gaussian distribution of ratings.

In our setting, we consider a set of users $U = \{u_1, \ldots, u_n\}$, a set of items $Y = \{y_1, \ldots, y_m\}$, and a set of possible ratings $V = \{v_1, \ldots, v_k\}$. In CF, we are interested in the conditional probability $P(v|u, y)$ that a user u will rate an item y with a value v. If the rating v is based on a numerical scale, it is appropriate to define the deterministic prediction function $g(u, v)$ that indicates the user's rating on the item as follows:

$$g(u, y) = \int_v v p(v|u, y)\, \mathrm{d}v \tag{1}$$

where $p(v|u, y)$ denotes a probability mass function (discrete case) or a conditional probability density function (continuous case) depending on the context. We introduce a variable z which can be treated as a group of items with similar features or attributes. Therefore, $p(v|u,y)$ can be calculated as

$$p(v|u, y) = \sum_z p(z|y) p(v|u, z) \tag{2}$$

Since items sharing similar preferences from users are clustered into one item community, most of the ratings from one user in the community will fall into the same rating range. We assume that the ratings from a certain user on all items in the community satisfy a Gaussian distribution:

$$p(v|u, z) = p(v; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(v-\mu)^2}{2\sigma^2}\right] \tag{3}$$

Therefore, the deterministic prediction function $g(u, v)$ that predicts the user's rating on item y can be computed as

$$g(u, y) = \int_v v p(v|u, y)\, \mathrm{d}v = \int_v \sum_z p(z|y) p(v|u, z) v\, \mathrm{d}v = \sum_z p(z|y) \int_v p(v|u, z) v\, \mathrm{d}v = \sum_z p(z|y) \mu_{y,z} \tag{4}$$

The user's rating on item y can be regarded as the sum of the product of $\mu_{y,z}$ (the average rating of user u in community z) and the posteriori probability $p(z|y)$ which depends on the relationship between item y and item community z.

In this model, whether item y is worth recommending is related with the posteriori probability $p(z|y)$. In order to estimate it, we can group the items into similar item communities and calculate the relationship as the estimation. When grouping the items for parameter estimation, we consider not only user rating data but also content information of items. Thus, it would help alleviate the problems caused by the sparseness of user rating data.

The probabilistic model we proposed for our CMRS can adopt various clustering algorithms to estimate the parameters. Although some advanced methods such as an incremental clustering algorithm can be potentially applied for a better performance and flexibility, we use a simple k-medoids (Han & Kamber, 2000) clustering method in the current work. The k-medoids algorithm makes each cluster to be represented by one of the objects in the cluster, whereas a cluster is represented by the centroid of all the member objects in the well-known k-means algorithm. Thus it is less influenced by outliers or other extreme values than the k-means algorithm. Both ratings and item attributes such as genre are used as features of the objects in our clustering work.

3. Parameter estimation

There are various ways to estimate the parameters. One of the ways to estimate the posteriori probability $p(z|y)$ is to use Bayes' rule.

$$p(z|y) = \frac{p(y|z)p(z)}{\sum_{z'=1}^{k} p(y|z')p(z')} \tag{5}$$

where $p(y|z)$ is the conditional probability of item y given the community z. It reflects the degree to which item y is associated with the community z, which can be evaluated by the Euclidean distance of two vectors. $p(z)$ is the ratio of the items falling into the community z over all items. To be more precise,

$$p(y|z) = \mathrm{ED}(V_y, V_z)^{-1}, \quad p(z) = \frac{|C_z|}{|C_Y|} \tag{6}$$

where V_y denotes the rating vector of item y and V_z denotes the center vector of variable z. C_z and C_y are the set of the community z and the set of all items, respectively. $\mathrm{ED}(\cdot)$ is a function of Euclidean distance between items. A shorter distance represents a closer relationship.

In the CF context, since users can only make partial ratings for one item, not all the elements of an item vector are filled in; some of them are missing and need to be predicted for recommendation. Based on our preliminary experiments, we chose to use an adjusted Euclidean distance in estimating and calculating the relationship of two items. When calculating the distance of two item vector, only the elements both vectors have non-zero values are used. In other words, ratings for two items being compared are considered for calculating the distance only when they are given by the same user.

$$\mathrm{ED}'(V_y, V_z) = \frac{\max(|V_y \cap V_z|, \beta)}{\beta} \mathrm{ED}(V_y, V_z) \tag{7}$$

where $|V_y \cap V_z|$ is the number of "common" ratings made by the same user, β is the threshold. By doing this, the items with more "common ratings" in calculating Euclidean distance have a high credit.

In calculating Euclidean distance, some of ratings are "real ratings" made by the users, some of ratings are "pseudo ratings" obtained from content information of items. As in the example shown in Table 1, the genre information of items can be numerated and treated as the "pseudo ratings." On one side they are the aggregate features of items, on the other side they are the "pseudo ratings" of the recommender system. When the "real ratings" of items are missing, especially for new items just resisted in the system, the "pseudo ratings" can be applied to get the relationship among items.

3.1. Creation of aggregate features

In this section, we describe the way of getting "pseudo ratings" or "aggregate features" for constructing item communities. To deal with the three challenges more effectively in CF in general and clustering in particular, we group items not only based on user ratings but also on the attributes of items. Our CMRS is developed for recommending ring tones for cell phone users. Most of ring tones are not tagged with abstract attributes such as genre information. It is also time-consuming and tedious to manually tag the genre information for each ring tone. Therefore, we cluster items (ring tones) based on the physical features such as timbral texture, rhythmic content and pitch content features to create some aggregate features, which can be treated more or less as the genre attribute.

The aggregate features provide abstract summarization of the primitive physical features, revealing the high-level semantic information. Based on our preliminary experiments, the high-level semantic information of items (genre as shown in Table 1) helps enhance recommendation. In addition, if we directly use primitive physical features for item community construction, all physical features are treated with equal importance in deciding the relationship among items in the current.

The clustering method to transfer physical features into aggregate features is derived from k-medoids Clustering Algorithm (Han & Kamber, 2000), which is extended in such a way that we apply the fuzzy set theory to represent the affiliation of objects with cliques (aggregate features). The probability of one object (here one object denotes a piece of music) assigned to a certain clique k is (see Fig. 1)

$$\mathrm{Pro}(j,k) = 1 - \frac{\mathrm{ED}(j,k)}{\mathrm{MaxED}(i,k)} \tag{8}$$

where the ED(j, k) denotes the Euclidean distance between object j and the centroid of clique k; Max ED(i, k) denotes the maximum Euclidean distance between an object and the centroid of clique k.

However, in our adjusted k-medoids algorithm, the fuzzy membership in a cluster is only assigned at the last step. It seems to unessentially represent the fuzzy memberships of objects. So the fuzzy k-means algorithm (Duda, Hart, & Stork, 2000) is also applied to group the items, where a fuzzy membership degree is assigned to each object in every iteration step as shown in Fig. 2. The global cost function, membership between an object and a cluster, and the mean value of one cluster are calculated as follows:

$$\begin{aligned}
\mathrm{GCF}_{\mathrm{fuz}} &= \sum_{i=1}^{\mathrm{c}} \left(\sum_{j=1}^{n} (\mathrm{Pro}_{i,j}^{\mathrm{b}} \times \mathrm{Dis}_{i,j}) \right) \\
\mathrm{Mean}_j &= \frac{\sum_{j=1}^{n} (\mathrm{Pro}_{i,j})^{\mathrm{b}} X_j}{\sum_{j=1}^{n} \mathrm{Pro}_{i,j}^{\mathrm{b}}} \\
\mathrm{Pro}_{i,j} &= \frac{\left(\frac{1}{\mathrm{Dis}_{i,j}}\right)^{\frac{2}{b-1}}}{\sum_{r=1}^{\mathrm{c}} \left(\frac{1}{\mathrm{Dis}_{i,j}}\right)^{\frac{2}{b-1}}}
\end{aligned} \tag{9}$$

where GCF_{fuz} denotes the fuzzy global cost function; c is the cluster number; b is a free parameter chosen to adjust the blending of different clusters; $\mathrm{Dis}_{i,j}$ is the Euclidean distance between the mean value of cluster i and object j; X_j is the vector of object j; and $\mathrm{Pro}_{i,j}$ refers to the membership between cluster i and object j.

Regardless of the clustering algorithms that are employed, the method of choosing the initial cluster center is a critical issue. In this regard, we use the refinement algorithm suggested by Bradley and Fayyad (1998).

Algorithm 1 : Adjusted K-means Clustering	***Algorithm 2 : Fuzzy K-means Clustering***
Input : the number of clusters k and item attribute features. Output: a set of k clusters that minimizes the squared-error criterion, and the probability of each item belonging to each cluster center are represented as a fuzzy set.	Input : the number of clusters k and item attribute features. Output: the membership of items belonging to each cluster center.
(1) Arbitrarily choose k objects as the initial cluster centers; (2) Repeat (a) and (b) until small change; (a) (Re) assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster; (b) Update the cluster means, i.e., calculate the mean value of the objects of each cluster; (3) Compute the possibility between objects and each cluster center.	(1) Initialize the parameters, and membership between objects and clusters; (2) Repeat (a) and (b) until global cost function has small change; (a) Recompute the mean value of each cluster; (b) Recompute the membership of each object; (3) Return the membership.

Fig. 1. Two algorithms for building aggregate features.

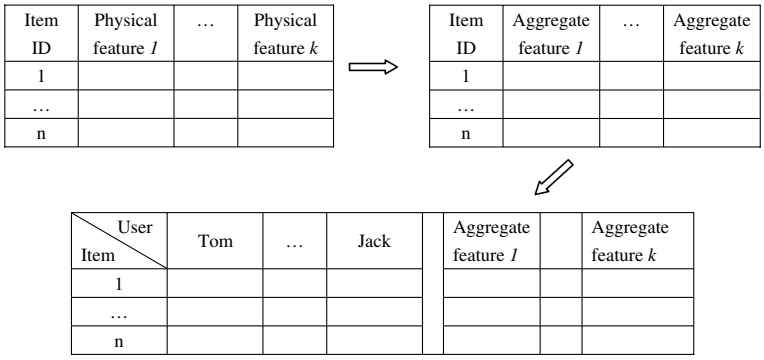

Fig. 2. Item community creation.

3.2. Item community creation

After getting the aggregate features of items, we attach those aggregate features to the user-rating matrix as shown in Fig. 2. As a result, we are able to build the item community based on both ratings and aggregate features.

Our clustering method for building community is derived from the well-known k-medoids clustering algorithm (Han & Kamber, 2000). Informally, our algorithm creates a fixed number, which is the number of communities, and then creates a composite object profile for each community as the representative of a community. The representative is a composite item that is the average of corresponding user ratings from the members of the community. The measure we use to compute the similarity between objects is the Pearson correlation coefficient. In effect, we are computing the extent to which two items are similar based on the extended rating matrix as shown in Fig. 2, which contains both user ratings and aggregate features. In addition, we do not consider the effect of negative correlations, because the clustering algorithm considered is not designed to handle negative similarities. Following is the detail of our clustering algorithm:

(1) Select k items at random to serve as initial cluster centers.
(2) Assign each object to the "best" cluster. An object's best cluster is defined to be the one whose center is best correlated with the object. The new center is the object that has the best overall correlation with all the other objects in the cluster. The Pearson correlation between an item and all of its other cluster members is applied as our measure of "best".

$$\mathrm{COR}_Z(k) = \sum_{l \in z \wedge k \neq l} \mathrm{sim}'(k,l) = \sum_{l \in z \wedge k \neq l} \frac{\mathrm{Max}(|I_k \cap I_l|, \gamma)}{\gamma} \mathrm{sim}(k,l) \tag{10}$$

where $\mathrm{COR}_z(k)$ denotes the correlation between the object k and its cluster z; $\mathrm{sim}(k,l)$ is the Pearson correlation between object k and l; $|I_k \cap I_l|$ is the number of common ratings made by the same user; γ is the threshold.
(3) Repeat steps 2 and 3 until the clusters are stabilized, i.e. no objects are assigned to new clusters.
(4) Create a representative of each cluster.

3.3. Feature extraction

Audio physical features extracted to create aggregate features should be helpful to distinguish music pieces. They can express some aspects of audio such as genre and rhythm. Our CMRS is designed for recommending ring tones for cellphone users, who usually select a ring tone by its rhyme. In this case, the aggregate features obtained from the vivid rhythmic, timbral and pitch content physical features are considered more meaningful to users than the textual description of genre, composers or singers.

Many of the previous reports on audio features are based on the retrieval of data from Musical Instrument Digital Interface (MIDI) corpora, where the files are symbolized by musical scores. However, considering the popularity of MPEG Layer 3 (MP3) digital music archives, we build our system based on audio features extracted from a MP3 music corpus.

Musical genre classification of audio signals has been studied by many researchers (Tzanetakis & Cook, 2002), which gave us some hints in selecting audio features to distinguish music objects in our CMRS. Three feature sets for representing timbral texture, rhythmic and pitch content were proposed, and the performance and relative importance of the proposed features was examined by our system using real-world MPEG audio corpus. We group the music based on those primitive physical features and the clusters obtained are treated as the aggregate features. The synopsis of used terms to describe these audio features is summarized in Table 2.

Timbral texture features are based on the standard features proposed for music-speech discrimination (Tzanetakis & Cook, 2002) as follows:

(1) Spectral centroid: It is the balancing point of the subband energy distribution and determines the frequency area around which most of the signal energy concentrates. To formulize

$$C(t) = \frac{\sum_{i=0}^{I-1}(i+1)s_i(t)}{\sum_{i=0}^{I-1}s_i(t)} \tag{11}$$

(2) Spectral rolloff point: It is used to distinguish voice speech from unvoiced music, which has a higher rolloff point because their power is better distributed over the subband range. To formulize

$$\sum_{i=0}^{R} s_i(t) = 0.85 \cdot \sum_{i=0}^{I-1} s_i(t) \tag{12}$$

(3) Spectral flux: It determines changes of spectral energy distribution of two successive windows. To formulize

$$\Delta(t, t+1) = \sqrt{\sum_{i=0}^{I-1}\left|\frac{s_i(t)}{\max(s_j(t) : 0 \leqslant j \leqslant I-1)} - \frac{s_i(t+1)}{\max(s_j(t+1) : 0 \leqslant j \leqslant I-1)}\right|^2} \tag{13}$$

(4) Sum of scale factor (SSF): the loudness distribution for whole duration.
(5) Mel frequency cepstral coefficients (MFCC): They are a set of perceptually motivated features and provide a compact representation of the spectral envelope such that most of the signal energy is concentrated in the first coefficients. The feature vector for describing timbral texture consists of the following features: means and variances of spectral Centroid, RollOff, Flux, SSF and first five MFCC coefficients over the texture window.

Rhythmic content features for representing rhythm structure are based on detecting the most salient periodicities of the signal. Gorge's method (Tzanetakis & Cook, 2002) is applied to construct the beat histogram, from which six features are calculated to represent rhythmic content.

(1) $A0$, $A1$: relative amplitude of the first and second histogram peak;
(2) RA: ratio of the amplitude of the second peak divided by the amplitude of the first peak;

Table 2
Synopsis of used terms

n	Time index 0 $nN-1$
i	Subband index 0 $iI-1$
Si(n)	Subband value at time index n for Subband i
m	Time position within window 0 $mM-1$
M	Analysis window size
t	Analysis window number

(3) $P1$, $P2$: period of the first, second peak in bpm;
(4) SUM: overall sum of the histogram (indication of beat strength).

Pitch content features describe the melody and harmony information about music signals and are extracted based on various pitch detection techniques (Tzanetakis & Cook, 2002). Basically the dominant peaks of the autocorrelation function, calculated via the summation of envelopes for each frequency band obtained by decomposing the signal, are accumulated into pitch histograms and the pitch content features are then extracted from the pitch histograms. The pitch content features typically include: the amplitudes and periods of maximum peaks in the histogram, pitch intervals between the two most prominent peaks, the overall sums of the histograms. Because the different magnitude of features, we mapped all audio features into [0 1].

4. Experimental evaluation

Experiments were carried out to examine the importance of using content information, beyond user ratings, in achieving better personalized recommendation. We further show how primitive audio features can be combined into aggregate features for the proposed CRMS and analyze their influences on recommendation performance. Although this model was developed originally for music collaborative recommendation based on audio features, our experiment with the movie data set demonstrates that it can be applied to other domains.

With these goals in mind, we used two data sets in our experiments. One is the Music data set constructed from a real-word music corpus that consists of sample files of mobile phone ring tones for campus users of Harbin Engineering University. It has 760 pieces of music and 4340 ratings made by 328 users. The music is stored as 22,050 Hz, 16-bit, mono MP3 audio files. The other is the EachMovie data set. It is courteously provided by DEC and has been widely used to evaluate the collaborative filtering techniques. There are 1623 movie items in this data set and 61,265 user profiles with a total of over 2.8 million ratings. The rating scale is discrete, taking values from 1 to 5. Note that the original ratings were between 0 and 1 and have been mapped to the new scale. Experiments on the EachMovie data set shows the adaptability of our approach to movies where the content information comes from the textual descriptions of movies such as genres, players, and directors.

4.1. Evaluation metrics

Mean absolute error (MAE) has widely been used in evaluating the accuracy of a recommender system (Breese et al., 1998). It is calculated by averaging the absolute errors in rating-prediction pairs as follows:

$$\mathrm{MAE} = \frac{\sum_{j=1}^{N} |P_{i,j} - R_{i,j}|}{N} \tag{14}$$

where $P_{i,j}$ is system's rating (prediction) of item j for user i, and $R_{i,j}$ is the rating of user i for item j in the test data. N is the number of rating-prediction pairs between the test data and the prediction result. A lower MAE indicates greater accuracy.

Typically, users show interests on the top-n (e.g. top-10) recommended items from the system. They would not like to go through a long recommendation list. Therefore, Breese et al. (1998) define the rank score as the expected utility of a ranked list of items, whose goal is to evaluate the performance based on not only the predication value but also the order of the recommended items. Formally,

$$R_{\mathrm{u}} = \sum_{j} \frac{\max(v - \bar{v}, 0)}{2^{(j-1)/(\alpha-1)}} \tag{15}$$

where j is the rank of an item in the full list of suggestions proposed by a recommender; $\bar{v}$ denotes the overall mean vote; α is the viewing half-life, which is the place of an item in the list such that it has a 50% chance of being viewed. As in their paper, we set it as 5, and found that our resulting conclusions were not sensitive to the precise value of this parameter. The final score reflecting the utilities of all users in the test set is

$$R = 100\frac{\sum_u R_u}{\sum_u R_u^{\max}} \tag{16}$$

where $R_u^{\max}$ is the maximum achievable utility if all observed items had been at the top of the ranked list, ordered by vote value. This transformation allows us to consider results independent of the size of the test set and number of items predicted in a given experiment. A higher rank scoring indicates greater accuracy.

Allbut1 (Breese et al., 1998) protocol was applied to evaluate the obtained prediction accuracies. More precisely, we randomly left out exactly one rating for every user who possesses at least two ratings. Notice that this uses somewhat less data than required, but allows us to use a single model to evaluate the leave-one-out performance averaged over all users. We have repeated the leave-one-out procedure 25 times with different random seeds. The reported numbers are the mean performance averaged over those runs.

4.2. Contribution of the content information

In this section, we examine the contribution of content information for personalized recommendation using our probabilistic model. In our music recommender system, we treated the primitive audio features of music as the content information. Three low-level physical feature sets for representing timbral texture, rhythmic and pitch content were considered to build the aggregate features for recommendation. In order to show the adaptability of our approach to other domains, we also tested on the movie data where the textual description of movies including genre, player and director information is applied to create the aggregate features.

Before examining the contribution of content information on both data sets, we investigate the factors to be optimized to obtain the optimal overall performance. These factors vary with the recommendation data and directly affect the overall performance. One is the community size and the other is the number of aggregate features.

4.2.1. Community size

An item community is the representative of objects (e.g. books and music) with the similar pattern. The number of community varies with the intrinsic item patterns. If the data set has more diverse item patterns, the number of community tends to increase. In our clustering phase, we classified the objects into groups. In the case that equals to 1, it means that all objects are treated as one community. In Fig. 3, it is observed that the number of communities affects the quality of prediction to a great extent, and the optimal performance is achieved with $k = 50$ and $k = 80$ for Music and EachMovie, respectively.

4.2.2. Number of aggregate features

The performance of our approach rests on the valuable information extracted from the content information of items such as textual description of genre and audio features. In our approach, we integrated the information by creating aggregate features. We implemented two clustering algorithms to build the aggregate features for further recommendation and test them with the different numbers of clusters on both the EachMovie and Music data sets. While we initially thought that the number of aggregate features would affect the performance, we found that the recommendation performance is not very sensitive to the number of aggregate features. As shown in Fig. 4, the fluctuation range is less than 0.01 and 0.02 for the large-scale data set case with textual information (EachMovie) and the small-scale data set with audio information (Music), respectively.

In addition, comparing with the adjusted k-means algorithm, the fuzzy k-means algorithm improves the performance a little. After the number of aggregate features reaches a certain threshold, its impact becomes fading when we apply the fuzzy k-means algorithm. Therefore, it is more reasonable to use the fuzzy k-means algorithm which is insensitive to the number of aggregate features while gaining a performance increase. In the subsequent experiments, we applied the fuzzy k-means algorithm and set the number of aggregate features to 40 and 60 for music and EachMovie data set, respectively.

4.2.3. Content information

In order to observe the contribution of content information in our model, we compare the performance of our approach under two different cases. One is applying our probabilistic model based on a user-rating matrix

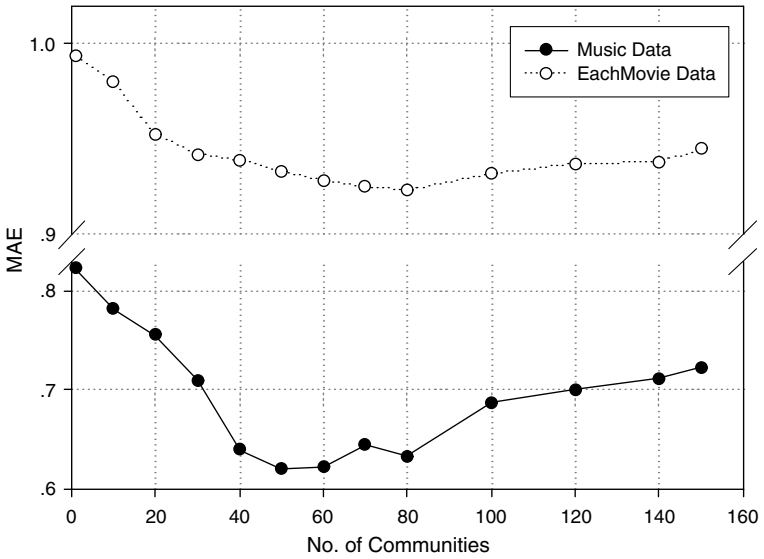

Fig. 3. Sensitivity of the community size.

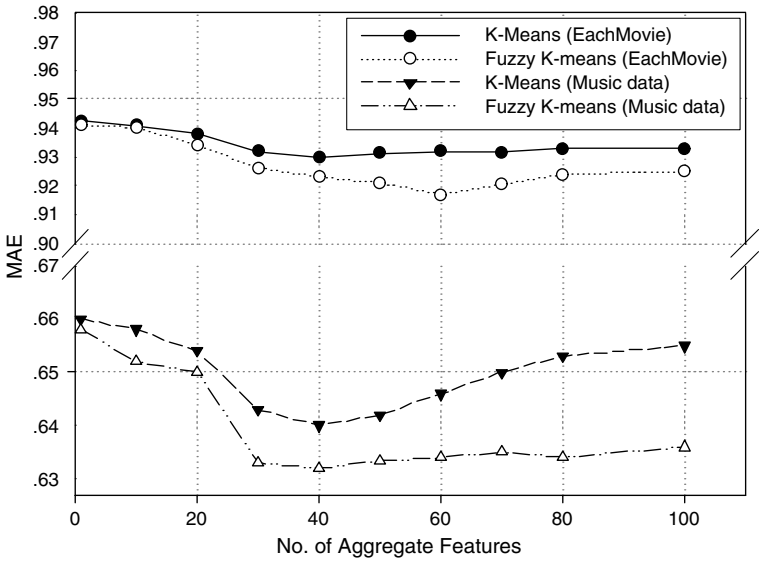

Fig. 4. Number of aggregate features.

with aggregate features (marked with "Aggregate Features" in Fig. 5). The other is using our probabilistic model based on a user-rating matrix without aggregate features (marked with"No Aggregate Features" in Fig. 5). As shown in Fig. 5, regardless of the types of the aggregate features, either created by textual information in the EachMovie data set or the audio features in the Music data set, they contributed to performance enhancement. We attribute such gain to the ability of our approach to apply content information to alleviate the three problems associated with data sparseness. Without content information, as shown in Table 1, Music 6 cannot get any predications due to the non-association problem. There are also no predications for Music 5 due to the cold start problem. Music 3 cannot have the privilege to be delivered to user Jack due to the user bias problem. In other words, our experiments proved that our approach can take the right utilization of content information to alleviate the three problems (user bias, non-association, and cold start problem) and consequently improve the recommendation performance.

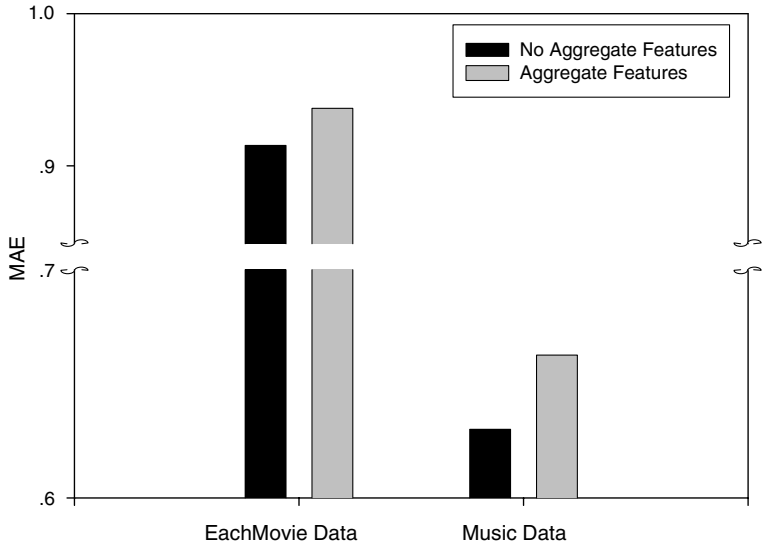

Fig. 5. Aggregate features versus no aggregate features.

Known that the effectiveness of utilizing the content information in our probabilistic model from the above experiment, we would like to examine the power of our approach employing the content information while comparing with other classic recommendation techniques. To this end, we implemented a standard memory-based method, a Simple Pearson method (Breese et al., 1998), and a standard item-based CF (Sarwar et al., 2001) method to be compared with our approach. As shown in Tables 3 and 4, our approach has a better performance than the others on both data sets though the improvements in MAE and rank score measure are different. These two different measures were used to emphasize on different evaluation aspects. MAE mainly evaluates the prediction precision while the rank scores put more emphasis on the rank of the recommended items. From the experimental result, more obvious improvement is observed from the rank than from the prediction precision.

4.2.4. Audio features for extracting content information

Since aggregate features help enhance the performance as shown in the above experiments, we would like to know what kinds of lower-level audio features are useful to build the aggregate features and their relative importance in our music recommender system. We carried out a series of experiments to this end. We constructed the aggregate features by individual audio physical features that are widely used for automatic clas-

Table 3
Comparison with the EachMovie data

Method	Rank scoring	Rel. improv. (%)	MAE	Rel. improv (%)
Pearson method	13.46	–	1.472	–
Item-based Pearson	21.50	59.7	0.984	33.1
Our approach	25.86	92.1	0.914	37.9

Table 4
Comparison with the music data

Method	Rank scoring	Rel. improv. (%)	MAE	Rel. improv (%)
Pearson Method	20.6	–	0.78	–
Item-based Pearson	27.4	33	0.69	11.5
Our approach	34.2	66	0.63	19.2

sification of music (Tzanetakis & Cook, 2002). As shown in Fig. 6, it is clear to observe that MFCC, RollOff, Centroid and Rhythmic are a little more effective than other feature sets. In other words, the timbral and rhythmic information is a little more helpful to distinguish ring tones of mobile phone than the loudness and pitch.

We then carried out a series of experiments with a combination of two audio features to construct the aggregate features. From these experiments, we found that the combination of MFCC and Rhythm achieved a better performance than other combinations. We further carried out our experiments using the combination of three audio features to construct the aggregate features. The conclusion is that the combination of MFCC, RollOff and Rhythm shows the best performance over all other combinations of three audio features, as shown in Fig. 7, where Nos. 1, 2 and 3 refer to MFCC, RollOff and Rhythm accordingly. In Fig. 7, the word

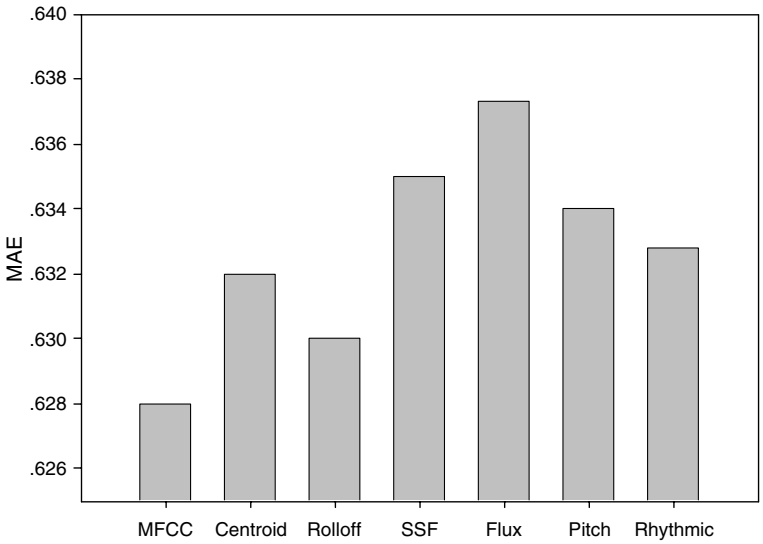

Fig. 6. Individual contributions of audio features.

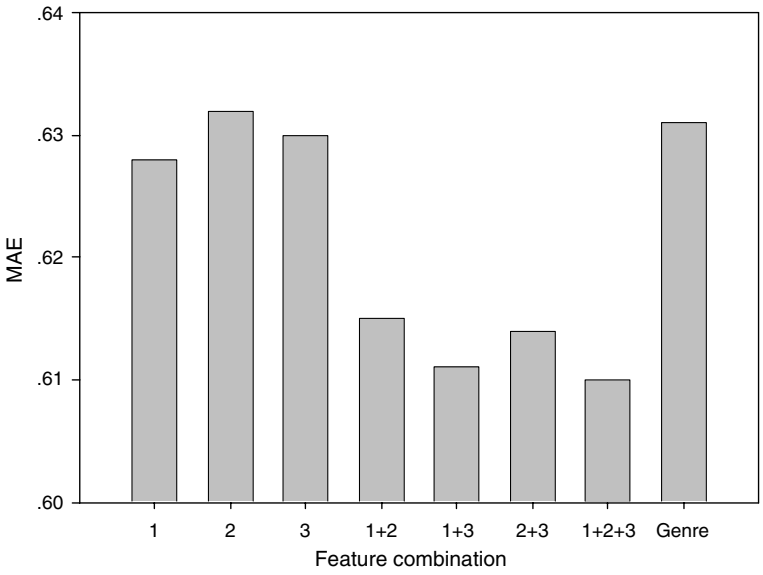

Fig. 7. Contribution comparison of audio features.

"Genre" represents that using the textual genre description for recommending the ring tone instead of the aggregate features of audio pieces. The aggregate features obtained from the physical features of audio pieces shows a better performance than the textual genre description. It reveals that mobile phone users select their preference not based on the genre information but on other physical features such as rhythm, timbre and pitch. By realizing this, our CMRS is designed to apply both physical features of music and user ratings for better personalized recommendation.

5. Conclusions

In this paper, we described a collaborative music recommender system for online mobile phone ring tones. It is based on our proposed item-based probabilistic model, where items are classified into groups or communities and predictions are made for users considering the Gaussian distribution of user ratings. This model provides a way of alleviating the three problems in similarity calculation of item, non-association, user bias from historical ratings, and cold start problems, by directly apply the physical audio features of music objects for personalized recommendation. Although our initial goal is to integrate the audio physical features for collaborative filtering, this model provides us to estimate the parameters so flexibly that make it possible to integrate various kinds content information of items such as textual and audio information.

Our experiments show that the proposed method outperforms the two other standard methods. In addition, we learned that the number of communities resulted from clustering is an important factor for effectiveness of the proposed method.

As described, our current work only focuses on the numerical ratings, however, binary ratings as millstreams is widely used on the internet and studied by many researchers we will extend our approach to the binary ratings based on the Bernoulli or Poisson models (Nasraoui & Pavuluri, 2004), which are more accurate than Gaussian model for binary data.

Acknowledgements

The EachMovie and Music data set were generously provided by Digital Equipment Corporation and Donghai Guan, respectively. This work was partially supported by Korea Research Foundation Grant (No. R05-2004-000-10190-0) and Post-doctoral Research Funds from ICU.

References

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

Bradley, P. S., & Fayyad, U. M. (1998). Refining initial points for *k*-means clustering. In *Proceedings of the ICML '98* (pp. 91–99).

Breese, J. S., Heckerman, D., & Kardie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th UAI* (pp. 43–52).

Chen, Hung-Chen & Arbee L. P. Chen. (2001). A music recommendation system based on music data grouping and user interests. In *Proceedings of the 10th CKIM*.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., & Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. In *Proceedings of the ACM-SIGIR workshop on recommender systems: algorithms and evaluation*.

Delgado, J., Ishii, N., & Ura, T. (1998). Content-based collaborative information filtering: actively learning to classify and recommend documents. In *Proceedings of the second international workshop, CIA'98* (pp. 206–215).

Duda, R. O., Hart, Peter E., & Stork, David G. (2000). *Pattern classification* (pp. 528–530). New York: Wiley-Interscience Publication.

Fisher, D., Hildrum, K., Hong, J., Newman, M., Thomas, M., & Vuduc, R. (2000). SWAMI: a framework for collaborative filtering algorithm development and evaluation. In *Proceedings of the SIGIR 2000* (pp. 366–368).

Ghias, A., Logan, J., Chamberlin, D., & Smith, BC. (1995). *Query by humming musical information retrieval in an audio database*. ACM multimedia.

Gupta, Dhruv, Mark Digiovanni, Hiro Narita, & Kenneth Y. Goldberg, Jester 2.0: Evaluation of an new linear time collaborative filtering algorithm. In *Proceedings of the SIGIR-99* (pp. 291–292).

Han, J., & Kamber, M. (2000). *Data mining: Concepts and techniques*. New York: Morgan-Kaufman.

Hauver, D. B. (2001). Flycasting: using collaborative filtering to generate a play list for online radio. In *Proceedings of the international conference on web delivery of music*.

Hofmann, T. (2003). Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proceedings of the SIGIR'03* (pp. 259–266).

Li, Qing, & Kim, B.M. (2003). Clustering approach for hybrid recommender system. In *Proceedings of the IEEE/web intelligence-2003*, Halifax, Canada.

Li, Qing., & Kim, B. M. (2004). Constructing user profiles for collaborative recommender system. *Proceedings of the Apweb2004. Lecture notes in computer science (LNCS)* (Vol. 3007, pp. 100–110). Springer.

Li, Qing, & Kim, B. M. (2005). Clustering for probabilistic model estimation for CF. In *Proceedings of the WWW 2005*.

Li, Qing, Kim, B. M., Guan, D. H., & Oh, D. H. (2004). A music recommender based on audio features. In *Proceedings of the SIGIR-04*, Sheffield, UK.

Li, Qing., Kim, B. M., Kim, J. W., & Kim, J. (2004). *A new collaborative recommender system addressing three problems. Lecture notes in artificial intelligence* (Vol. 3157). Springer.

Nasraoui, O., & Pavuluri, M. (2004). Accurate web recommendations based on profile-specific URL-predictor neural networks. In *Proceedings of the WWW 04* NY.

O'Conner, M., & Herlocker, J. (1999). Clustering items for collaborative filtering. In *Proceedings of the ACM-SIGIR workshop on recommender systems*.

Resnick, P., Iacovou, N., Suchak, M., Bergstorm, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of Netnews, In *Proceedings of the ACM CSCW-94* (pp. 175–186).

Sarwar, B. M., Karypis, G., Konstan, J. A., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international WWW conference 2001* (pp. 285–295).

Shardanand, U., & Maes, P. (1995). Social information filtering: algorithms for automating "Word of Mouth. In *Proceedings of the ACM CHI'95 conference on human factors in computing systems* (pp. 210–217).

Tzanetakis, G., & Cook, P. (2002). Music genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing, 10*(5).

Wasfi, A. M. A. (1999). Collecting user access patterns for building user profiles and collaborative filtering. In *Proceedings of the IUI* (pp. 57–64).

Wittenburg, K., Das, D., Hill, W., & Stead, L. (1995). Group asynchronous browsing on the world wide web. In *Proceedings of the fourth WWW* (pp. 51–62).