Personalized Next-song Recommendation in Online Karaokes

Xiang Wu[1], Qi Liu[1], Enhong Chen[1], Liang He[1] Jingsong Lv[2], Can Cao[2], Guoping Hu[2]
[1]School of Computer Science and Technology University of Science and Technology of China
[2]Anhui USTC iFLYTEK Co., Ltd., China
wux@mail.ustc.edu.cn, {qiliuql,cheneh}@ustc.edu.cn
hshl05@mail.ustc.edu.cn {jslv, cancao, gphu}@iflytek.com

ABSTRACT

In this paper, we propose Personalized Markov Embedding (PME), a next-song recommendation strategy for online karaoke users. By modeling the sequential singing behavior, we first embed songs and users into a Euclidean space in which distances between songs and users reflect the strength of their relationships. Then, given each user's last song, we can generate personalized recommendations by ranking the candidate songs according to the embedding. Moreover, PME can be trained without any requirement of content information. Finally, we perform an experimental evaluation on a real world data set provided by ihou.com which is an online karaoke website launched by iFLYTEK, and the results clearly demonstrate the effectiveness of PME.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*

Keywords

Music Recommendation; Personalization; Markov Embedding

1. INTRODUCTION

Advances in the Internet and mobile technologies have exposed people to the massive amount of online multimedia entertainment. Among them, online karaoke has attracted significant attention, since everybody can sing along with the recorded music anytime and anywhere, or even with a music video like a professional singer. As a trend, much more music are becoming available, and thus the demand for intelligent karaoke services, i.e., personalized song recommendation, is expected to increase dramatically.

Actually, music recommendations have been widely studied and applied in the literature. For instance, since music is rich in both textual and acoustic information (e.g., artists, genres and pitches), several recommendation algorithms[4, 8, 12, 15] exploited it and modeled users or user behaviours based on extracted features. However, content information cannot be easily extracted in many cases. Thus, collaborative filterings have become the most popular music recommendation techniques. Methods such as neighborhood based methods[14], matrix factorization[9] and Markov model[5] are commonly adopted. These technical achievements further secured the success of many music station websites, such as Pandora[1] and douban.fm[2]. In summary, most of the existing works try to predict the possible ratings from a given user to each song (or other multimedia items such as videos and movies), and then generate the candidate songs (or videos) for users to listen to (or to watch). However, the above strategies cannot be directly applied to the next-song recommendation for karaoke users due to the domain and technical challenges.

Figure 1: A screenshot of ihou.com, where a user is singing.

To illustrate the first challenge, we will take ihou.com[3] as an example. ihou.com is a famous online karaoke website launched by iFLYTEK[4], the leading provider of Chinese speech and language technology. Figure 1 is a screenshot of ihou.com, where a user is singing a Chinese song and just got an 80 out of 100 for the latest performed sentence. Thus, different from traditional online entertainment (e.g., music listening and movie watching[10]) where users passively experience items and give ratings to the items and systems, the users in karaoke systems like ihou.com are usually being evaluated by the system, based on their singing performances. This leads to the challenge that we cannot refer to the explicit user ratings for representing user preferences. However, we could assume that every choice of a song indicates a strong interest of the karaoke user in that music, since the user has to spend minutes performing himself and meanwhile take the risk of being judged.

The second challenge lies in the sequential behavior of users's singing[11]. Similar to web searches[3], where user's preceding queries can help to capture his/her search intents, the recent choices of karaoke users also reflect their current moods and interests[2]. For instance, it is not likely for you to sing *"My Heart Will Go On"*,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
RecSys'13, October 12–16, 2013, Hong Kong, China.
Copyright 2013 ACM 978-1-4503-2409-0/13/10 ...$15.00.
http://dx.doi.org/10.1145/2507157.2507215.

[1]http://www.pandora.com
[2]http://douban.fm
[3]http://www.ihou.com/
[4]http://www.iflytek.com/english/

when the cheerful *"Uptown Girl"* is just sung, even if you do like these two songs. Thus, besides capturing the general (long-term) preference of each user, it is also important to model the sequential nature of singing lists for the user's contextual (short-term) preference[13] when making personalized next-song recommendations.

To address the above challenges, in this paper, we extend the existing Logistic Markov Embedding (LME)[5] algorithm and propose Personalized Markov Embedding (PME), a next-song recommendation strategy for online karaoke users. Specifically, we first embed songs and users into a Euclidean space in which distances between songs and users reflect the strength of their relationships. This embedding could efficiently combine users' long-term and short-term preferences together. Then, given each user's last song, we can generate recommendations by ranking the candidate songs according to the embedding. Moreover, our PME can be trained without any content information of songs, namely just with the interaction history of users' singing. Finally, we perform an experimental evaluation on a real world data set, and the results demonstrate that the PME algorithm outperforms several state-of-the-arts, including the non-personalized LME algorithm.

2. NEXT-SONG RECOMMENDATION

2.1 Problem Formalization

Given the songs that have been performed previously by each karaoke user, our goal is to recommend a ranked list of songs to a given user for the current context (the last song). We formalize this problem as follows. Let $S = \{s_1, s_2, \cdots, s_{|S|}\}$ be the song set and $U = \{u_1, u_2, \cdots, u_{|U|}\}$ be the user set. The neighbouring song records, which are usually similar, performed by the same user during short time intervals form a session. For instance, the j-th session of user u is represented by $p_{u,j} = \left(p_{u,j}^{(1)}, p_{u,j}^{(2)}, \cdots, p_{u,j}^{(|p_{u,j}|)}\right)$ where $p_{u,j}^{(k)}$ is the k-th song in session $p_{u,j}$. All sessions produced by user u is $p_u = \left(p_{u,1}, p_{u,2}, \cdots, p_{u,|p_u|}\right)$. Thus, the entire data set can be represented as $D = \{(u, p_u) | u \in U\}$. In other words, given D, the current user u and the last song s performed by user u, we want to train a model to generate a candidate song list for user u to choose from for his/her next performance. Along this line, we should estimate the transition probabilities between songs for each user (session) by measuring user preferences.

2.2 Personalized Markov Embedding

In this subsection, we describe the way to simultaneously measure users' long-term and short-term preferences and the relationships between users and songs by PME. Thus, the next-song recommendation list can be generated naturally.

To this end, we map songs and users into points in a $\mathbb{R}^d$ space, and use vector $\mathbf{x}(s)$ and vector $\mathbf{y}(u)$ to denote song s's and user u's coordinates in this space, respectively. The Euclidean distance between two points reflects the relation between corresponding users/songs. If two songs stay apart from each other, it is not likely that they will show up in the same session. Also, if a user stays close to a song, he/she may often sing it. Worth noting that the relations are asymmetric, and this can be inferred from the later illustration.

First, we assume the probability $Pr(s_b|s_a, u)$ of a transition from song s_a to song s_b made by user u is related to the Euclidean distances $\|\mathbf{x}(s_a) - \mathbf{x}(s_b)\|_2$ and $\|\mathbf{y}(u) - \mathbf{x}(s_b)\|_2$, which can be viewed as user u's short-term and long-term preferences. Straightforwardly, it can be described as:

$$Pr(s_b|s_a, u) \propto e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s_b)\|_2^2 - \|\mathbf{y}(u)-\mathbf{x}(s_b)\|_2^2} \tag{1}$$

Note that $Pr(s_b|s_a)$ is proportional to $e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s_b)\|_2^2}$ in LME[5]. Thus, the user information is ignored by LME, while being considered by our PME.

Since $\sum_{b=1}^{|S|} Pr(s_b|s_a, u) = 1$, we add a denominator to the exponential value for normalization, and Equation (1) becomes:

$$Pr(s_b|s_a, u) = \frac{e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s_b)\|_2^2 - \|\mathbf{y}(u)-\mathbf{x}(s_b)\|_2^2}}{\sum_{s\in S} e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s)\|_2^2 - \|\mathbf{y}(u)-\mathbf{x}(s)\|_2^2}} \tag{2}$$

And now, we can get the coordinate mappings through a likelihood maximization approach:

$$(\mathbf{X}, \mathbf{Y}) = \arg\max_{\mathbf{X},\mathbf{Y}} \prod_{(u,p_u)\in D} \prod_{j=1}^{|p_u|} \prod_{k=2}^{|p_{u,j}|} Pr\left(p_{u,j}^{(k)} \middle| p_{u,j}^{(k-1)}, u\right) \tag{3}$$

where the song-mapping matrix $\mathbf{X} = [\mathbf{x}(s_1), \mathbf{x}(s_2), \cdots, \mathbf{x}(s_{|S|})]$ and the user-mapping matrix $\mathbf{Y} = [\mathbf{y}(u_1), \mathbf{y}(u_2), \cdots, \mathbf{y}(u_{|U|})]$.

Then, we could transform Equation (3) into its equivalent form by applying the ln function:

$$\begin{aligned}
(\mathbf{X}, \mathbf{Y}) &= \arg\max_{\mathbf{X},\mathbf{Y}} \sum_{(u,p_u)\in D} \sum_{j=1}^{|p_u|} \sum_{k=2}^{|p_{u,j}|} \ln Pr\left(p_{u,j}^{(k)} \middle| p_{u,j}^{(k-1)}, u\right) \\
&= \arg\max_{\mathbf{X},\mathbf{Y}} \sum_{u\in U} \sum_{s_a\in S} \sum_{s_b\in S} c_{u,s_a,s_b} \ln Pr(s_b|s_a, u) \\
&= \arg\max_{\mathbf{X},\mathbf{Y}} \sum_{u\in U} \sum_{s_a\in S} \sum_{s_b\in S} c_{u,s_a,s_b} \Bigg[-\|\mathbf{x}(s_a) - \mathbf{x}(s_b)\|_2^2 \\
&\quad - \|\mathbf{y}(u) - \mathbf{x}(s_b)\|_2^2 - \ln \sum_{s\in S} e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s)\|_2^2 - \|\mathbf{y}(u)-\mathbf{x}(s)\|_2^2} \Bigg] \\
&\stackrel{\text{def}}{=} \arg\max_{\mathbf{X},\mathbf{Y}} L_1(D|\mathbf{X}, \mathbf{Y})
\end{aligned} \tag{4}$$

where c_{u,s_a,s_b} is the number of occurrence of song s_b after song s_a by user u in the whole data set D. While both $\frac{\partial L_1(D|\mathbf{X},\mathbf{Y})}{\partial \mathbf{x}(i)}$ and $\frac{\partial L_1(D|\mathbf{X},\mathbf{Y})}{\partial \mathbf{y}(i)}$ are non-convex, we find that gradient descent algorithm can still find proper solutions. However, both of the partial derivatives are so complex that the time complexity of a single iteration is as high as $O\left(|U||S|^2\right)$ even after optimization.

Then, to overcome the time-consuming problem, we propose Equation (5) to simulate Equation (2). In this way, the two types of Euclidean distances can be decoupled:

$$Pr(s_b|s_a)\, Pr(s_b|u) = \frac{e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s_b)\|_2^2}}{\sum_{s\in S} e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s)\|_2^2}} \frac{e^{-\|\mathbf{y}(u)-\mathbf{x}(s_b)\|_2^2}}{\sum_{s\in S} e^{-\|\mathbf{y}(u)-\mathbf{x}(s)\|_2^2}} \tag{5}$$

where $Pr(s_b|s_a)$ is the transition probability from song s_a to song s_b and $Pr(s_b|u)$ is the probability of user u singing song s_b. Note that Equation (5) is not simply an assembled model, since all parameters will be trained simultaneously.

Following a similar process of Equation (3) and Equation (4), we could get:

$$\begin{aligned}
(\mathbf{X}, \mathbf{Y}) &= \arg\max_{\mathbf{X},\mathbf{Y}} \sum_{u\in U} \sum_{s_a\in S} \sum_{s_b\in S} c_{u,s_a,s_b} \ln Pr(s_b|s_a)\, Pr(s_b|u) \\
&= \arg\max_{\mathbf{X},\mathbf{Y}} \sum_{u\in U} \sum_{s_a\in S} \sum_{s_b\in S} c_{u,s_a,s_b} \Bigg[-\|\mathbf{x}(s_a) - \mathbf{x}(s_b)\|_2^2 \\
&\quad - \ln \sum_{s\in S} e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s)\|_2^2} - \|\mathbf{y}(u) - \mathbf{x}(s_b)\|_2^2 \\
&\quad - \ln \sum_{s\in S} e^{-\|\mathbf{y}(u)-\mathbf{x}(s)\|_2^2} \Bigg] \\
&\stackrel{\text{def}}{=} \arg\max_{\mathbf{X},\mathbf{Y}} L_2(D|\mathbf{X}, \mathbf{Y})
\end{aligned} \tag{6}$$

It can be found that the time complexity of one iteration has decreased to $O(|U||S|)$, if $\sum_{s\in S} e^{-\|\mathbf{x}(s_a)-\mathbf{x}(s)\|_2^2}$ and $\sum_{s\in S} e^{-\|\mathbf{y}(u)-\mathbf{x}(s)\|_2^2}$ are cached for approximate calculation.

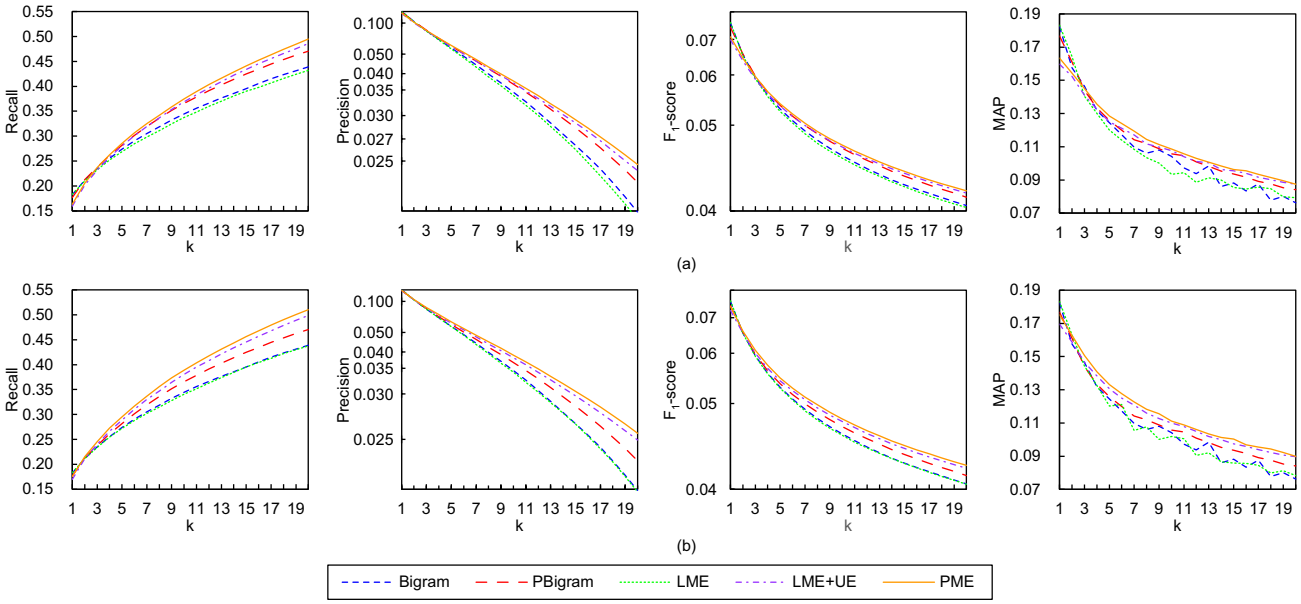

Figure 2: Top-k comparisons of Bigram, PBigram, LME, LME+UE and our PME in $\mathbb{R}^{20}$ (a) and $\mathbb{R}^{50}$ (b) with $k \in [1, 20]$.

Further, we apply regularization of Frobenius norm to Equation (6), and our target becomes:

$$(\mathbf{X}, \mathbf{Y}) = \arg\max_{\mathbf{X},\mathbf{Y}} \left[L_2\left(D\,|\mathbf{X}, \mathbf{Y}\right) - \lambda \|\mathbf{X}\|_{Frob}^2 - \lambda \|\mathbf{Y}\|_{Frob}^2 \right] \tag{7}$$

where λ is the regularization coefficient, and we can get the updating rules through partial derivations:

$$\mathbf{x}(s) \leftarrow \mathbf{x}(s) + \frac{\tau}{n}\left[\frac{\partial L_2\left(D\,|\mathbf{X}, \mathbf{Y}\right)}{\partial \mathbf{x}(s)} - \frac{\partial \lambda \|\mathbf{X}\|_{Frob}^2}{\partial \mathbf{x}(s)} \right] \tag{8}$$

$$\mathbf{y}(u) \leftarrow \mathbf{y}(u) + \frac{\tau}{n}\left[\frac{\partial L_2\left(D\,|\mathbf{X}, \mathbf{Y}\right)}{\partial \mathbf{y}(u)} - \frac{\partial \lambda \|\mathbf{Y}\|_{Frob}^2}{\partial \mathbf{y}(u)} \right] \tag{9}$$

where τ is the learning rate and n is the total number of song transitions in data set D.

Finally, after the embedding, given the current user u and his/her last song s_a, we could generate the next-song recommendation by ranking the candidate songs based on Equation (5). Meanwhile, we can figure out why $Pr(s_b|s_a, u)$ and $Pr(s_a|s_b, u)$ are not symmetric.

3. EXPERIMENTAL RESULTS

Table 1: Statistics of the data set.

#Users	#Items	#Sessions	#Training Transitions	#Test Transitions
13,452	943	105,743	332,640	58,687

In this section, we evaluate PME on the real world karaoke data provided by ihou.com from July 2011 to April 2012. To reduce noise, we only consider the users who have sung more than 10 different songs and the songs which have been sung by more than 3 different users. For the singing session segmentation, we let the songs to be in the same session, if the user's inactive intervals between adjacent songs are less than 1 hour. We put the last transition of songs from each session into the test set, the rest for training, and ensure that every test song exists in the training set. The statistical information of the final data can be found in Table 1.

Evaluation Metrics. To measure the ranking accuracy, we adopt Precision, Recall, F_1-score and Mean Average Precision (MAP) as our evaluation metrics[7]. These metrics pay more attention to the first several candidates in the ranked list, and try to characterize the recommendation results from different perspectives.

Baselines. We choose the following four baseline methods:

- **Bigram Model (Bigram)**[1] is a first-order Markov model which considers the probability of the appearance of songs separately for different preceding songs, in other words, it calculate $Pr(s_b|s_a)$ based on statistics. We adopt Witten-Bell discounting [6] for smoothing.
- **Personalized Bigram Model (PBigram)** is an assembled algorithm which combines two bigrams $Pr(s_b|s_a)$ and $Pr(s_b|u)$ together by multiplying their results, and Witten-Bell discounting is also used.
- **Logistic Markov Embedding (LME)**[5] is similar to PME. However, it outputs the same $Pr(s_b|s_a)$ for all users.
- **LME with User Embedding (LME+UE)** includes personalization into LME by a two-phased training approach, where song-embeddings are first trained and fixed before the training of user-embeddings are conducted.

All of the aforementioned four baselines can be seen as the related methods for PME. Worth noting that similar to bigram model, uniform model and unigram model [5] are also popular models for Natural Language Processing. However, their performances are no better than the selected baselines.

Performance Comparison. Parameters such as τ and λ in certain methods are determined experimentally, and all iterative methods are run until convergence. The performances under different dimensionalities are tested for LME, LME+UE and PME. For instance, the results of models trained in $\mathbb{R}^{20}$ and $\mathbb{R}^{50}$ are shown in Figure 2. From both Figure 2(a) and Figure 2(b), we could observe that our personalized methods (PBigram, LME+UE and PME) are better than their corresponding benchmark methods (Bigram and LME). More importantly, PME performs best in all cases. Another interesting observation is that LME performs very similarly to Bigram under the ranking metrics, which is different from that in [5] where the likelihood metric is adopted.

Also, we compare the efficiency difference between LME and PME. For instance, under the same platform, when the dimensionality equals to 50, it takes 1.3s for LME and 16.5s for PME to run a single iteration, whose time complexities are $O\left(|S|^2\right)$ and $O\left(|U|\,|S|\right)$, respectively.

Deep Understanding. For comparing the performance of the algorithms under different sparsity, we conduct another experiment.

Specifically, we first separate transitions in the test set into 5 different splits according to their occurrence time in the training data, with each split having roughly the same amount of transitions. For instance, the transitions in the test set that didn't occur or occurred only once in the training set and transitions in the test set that occurred 2 to 5 times are evaluated separately. We calculate the Recall value on Top-10 recommendations of three typical algorithms (Bigram, LME and PME) for all 5 splits in $\mathbb{R}^{50}$, and the final results are shown in Figure 3.

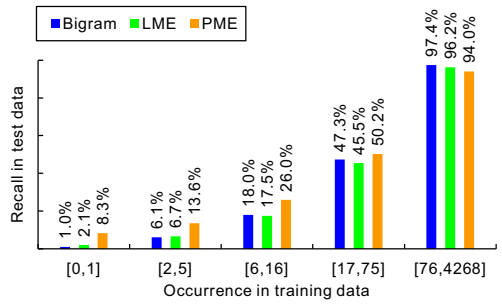

Figure 3: Comparisons of the Recall value on Top-10 recommendations of Bigram, LME and PME with different splits.

From Figure 3 we can see that for the transitions that don't occur many times (less than 17) in the training set, PME performs much better than Bigram and LME. However, the advantage of PME becomes smaller with the increase of the occurrence. Specifically, PME achieves an average lead of 7.6% over the first three ranges, which take up about 60% percent of the test data, and Bigram is slightly better than PME only on the last range. Since most of the transitions have comparably low occurrence rate, PME could outperforms LME and Bigram in real applications, i.e., PME is better at predicting the unseen and sparse data.

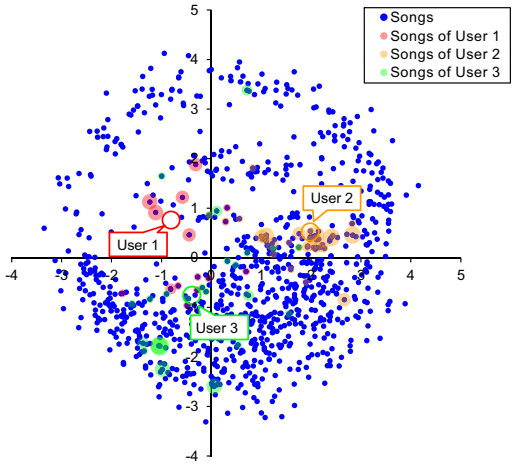

Figure 4: Visualization of PME in $\mathbb{R}^2$.

Case Study. Figure 4 is a visualization of the trained PME model in $\mathbb{R}^2$ where all songs are represented by blue dots and 3 randomly picked users are represented by circles with different colors. We find out the songs sung by these 3 users in the training set, and then highlight them by semitransparent circles with their sizes proportional to the singing frequency of the corresponding user.

It shows that PME can successfully extract user preferences, since the embedding position of each user is near to his/her previous actions. In the stage of recommendation, the songs that are close to the given user, or in other words close to the songs which the given user have sung, are more likely to be recommended to this user according to Equation (5).

4. CONCLUSIONS

In this paper, we presented Personalized Markov Embedding, a next-song recommendation strategy for online karaoke users. We first proposed to embed the songs and users into a Euclidean space in which distances between songs and users reflect the strength of their relationships. We also used a simulation to ensure that the embedding process can converge in time. In this way, the short-term and long-term preferences of each user can be captured without requiring any content information. Then, the next-song recommendations are conducted according to the embedding. Finally, the performance of the proposed PME are evaluated on a real world data set, and the experimental study delivers encouraging results.

5. ACKNOWLEDGEMENTS

The work was supported by grants from Natural Science Foundation of China (Grant No. 61073110), the Key Program of National Natural Science Foundation of China (Grant No. 60933013), Research Fund for the Doctoral Program of Higher Education of China (20113402110024), and the National Major Special Science and Technology Project (Grant No. 2011ZX04016-071).

6. REFERENCES

[1] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

[2] R. Cai, C. Zhang, C. Wang, L. Zhang, and W.-Y. Ma. Musicsense: contextual music recommendation using emotional allocation modeling. In *Proc. of MM'07*, pages 553–556. ACM, 2007.

[3] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click-through and session data. In *Proc. of KDD'08*, pages 875–883. ACM, 2008.

[4] H.-C. Chen and A. L. Chen. A music recommendation system based on music data grouping and user interests. In *Proc. of CIKM'01*, volume 5, pages 231–238, 2001.

[5] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims. Playlist prediction via metric embedding. In *Proc. of KDD'12*, pages 714–722. ACM, 2012.

[6] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–393, 1999.

[7] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM TOIS*, 22(1):5–53, 2004.

[8] K. Hoashi, K. Matsumoto, and N. Inoue. Personalization of user profiles for content-based music retrieval based on relevance feedback. In *Proc. of MM'03*, pages 110–119. ACM, 2003.

[9] N. Koenigstein, G. Dror, and Y. Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Proc. of RecSys'11*, pages 165–172. ACM, 2011.

[10] Q. Liu, E. Chen, H. Xiong, C. H. Ding, and J. Chen. Enhancing collaborative filtering by user interest expansion via personalized ranking. *IEEE TSMC-B*, 42(1):218–233, 2012.

[11] K. Oku, S. Nakajima, J. Miyazaki, S. Uemura, and H. Kato. A recommendation method considering users' time series contexts. In *Proc. of ICUIMC'09*, pages 465–470. ACM, 2009.

[12] E. Pampalk. *Computational models of music similarity and their application in music information retrieval*. 2006.

[13] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proc. of WWW'10*, pages 811–820. ACM, 2010.

[14] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proc. of CSCW'94*, pages 175–186. ACM, 1994.

[15] M. Tiemann and S. Pauws. Towards ensemble learning for hybrid music recommendation. In *Proc. of RecSys'07*, pages 177–178. ACM, 2007.