Signal Processing ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Music recommendation using graph based quality model

Kuang Mao [a,*], Gang Chen [a], Yuxing Hu [b], Luming Zhang [a]

Q1

[a] College of Computer Science, Zhejiang University, Hangzhou, China
[b] School of Aerospace, Tsinghua University, Beijing, China

ARTICLE INFO

Article history:
Received 14 August 2014
Received in revised form
10 October 2014
Accepted 31 March 2015

Keywords:
Music recommendation
Quality model
Graph based ranking

ABSTRACT

Nowadays, listening to music has become a habit for almost everyone. Music recommendation helps the users discover the songs they like to listen. In this paper, we propose a music recommendation framework based on graph based quality model to make fine-grained music recommendation. We first discover the recommendation cues, which we called user's preference relations from the users' ratings. Then we model them using the quality model and propose a regularization framework to calculate the recommendation probability of songs. Our experiments show that the proposed framework is superior to two traditional algorithms, especially in solving the cold start problem in recommendation.

Published by Elsevier B.V.

1. Introduction

Nowadays listening to music is a habit for most people. The prosperity of the popular music sharing social media websites such as Last.fm,[1] Pandora[2] provides us a way to discover the music. One important service of the social media is to provide song recommendations to the users. The general purpose of the song or track[3] recommendation is to help the user discover songs which they will prefer to listen. It is a necessary component of any music community and is also a way to promote the user to be active in the social community.

The early stage of music recommendation is based on analyzing the similarity between different songs which is called content based music recommendation [1–4]. These techniques use low level features to represent user's preferences of the songs such as moods and rhythms and recommend songs that have a large similarity on low level features. However, it suffers from the semantic gap between low level features and user's preferences. Modern music recommendation focuses on choosing song tracks from other related users' listening histories as recommendation. The most typical methods are Collaborative Filtering (CF) based methods [5–8]. These approaches exploit the similarity of different users based on users' interests and recommend songs among the similar users. But they do not explore whether the user will satisfy by recommending these songs. Another drawback is that the startup community will face severe cold start problem. Lacking user's listening history, the algorithms are not always working.

In the current music community, user rating is a valuable indication of users' preferences towards different music tracks. The user will give a high rating for the song they prefer. Based on the rating, we are able to discover listeners' choice on which song is more attractive comparing with another song. If many listeners share the same choice on a specific song pair, we can know which song would be more attractive to the user. Our recommendation aims to recommend more attractive songs comparing with users' listening history.

In this paper, we report a new song recommendation framework which discovers the attractiveness of each track to the user and make a fine-grained recommendation. Specifically, our recommendation framework takes a user's listening

* Corresponding author.
E-mail address: mbill@zju.edu.cn (K. Mao).
[1] http://www.last.fm.
[2] http://www.pandora.com.
[3] We use music, song, track interactively.

http://dx.doi.org/10.1016/j.sigpro.2015.03.026
0165-1684/Published by Elsevier B.V.

history as input, and predicts the songs that user will like to listen. Our song recommendation framework is different with traditional recommendation which focuses on discovering recommendation songs from the users who share the similar interests, we discover song's internal attractiveness relations which ensures that the recommended songs are more fair-sounding than user's listening history.

This song recommendation framework faces two challenges. The first one is that we need to decide which song is more attractive than another song. There exists no such kind of knowledge, we need to discover the attractiveness information from the music community. We proposed to discover the preference relation from the user's rating. The *preference relation* indicates which song is more attractive comparing with another song. Here more attractive means a song is easier to get a high user rating. We will implicitly discover the preference relation from user's song rating data. However, do the discovered preference relations fit every user's taste? We need to measure whether the preference relation is widely accepted by the users in the community. We propose to measure the reliability of the preference relation. Those with high reliability will be more important in the recommendation.

The second challenge is how to make song recommendation based on the discovered preference relations. Because we only have a limited number of preference relations, we need to model them so that we can make full use of them. In this paper, we propose to model the preference relation using a *preference graph*. We consider whether a song is attractive using a quantity called *preference probability*. We also design a song ranking algorithm through a regularization framework to calculate the preference probability of each track.

The advantage of our framework is that we overcome the cold start problem which is one of the biggest challenges in recommendation system. Another advantage is that once our model is built, it can be used in other music recommendation communities. Different community can also maintain a shared preference relation database to be used in recommendation.

We summarize our contribution of the paper as follows:

(1) We propose a new song recommendation framework to do a fine-grained song recommendation based on analyzing the attractiveness of the songs.
(2) We propose a song preference relation discovering strategy to mine users' preference towards different songs from the user ratings.
(3) We propose a preference graph to model the discovered preference relation and design a song ranking algorithm to do song recommendation.
(4) Our experiment shows that the proposed framework is superior to the traditional song recommendation algorithms.

The rest of the paper is organized as follows: Section 2 is the related work and Section 3 gives an overview of the system. In Sections 4 and 5, we give the details of the proposed framework. In Section 6, we demonstrate our experiments and the results.

2. Related work

This paper is related with the traditional song recommendation, graph based ranking algorithm. We will give a brief introduction of these works.

2.1. Song recommendation

Song recommendation aims at discovering songs that satisfy user's listening interests. The early stage of the methods is acoustic-based song recommendation [1–4]. It focuses on narrowing the semantic gap between low level features and high level music concepts [9]. They use low level features to represent user's preferences of the songs such as moods and rhythms and recommend songs that have a large similarity on low level features. However, this kind of representation cannot precisely represent the concepts.

After that, researchers focus on discovering useful information from user's ratings. They utilize a technique called Collaborative Filtering (CF) [10] to find related songs from other users' listening history. It can also be divided into user-based CF [11] and item-based CF [12]. Take the user-based CF as an example, it first predicts the listener's possible ratings for unrated items from a group of singers who share the similar listening histories. Then we rank the songs based on the predicted ratings. However, these methods will suffer from the cold start problem. With few user's listening history, it is almost not working.

Another popular way is to use graph based method to do song recommendation. The general idea is as follows. It is first to build a user-item bi-graph and then do random walk on the graph to make song recommendation. The structure of the graph makes sure those songs that share many connected users with the query song will get higher ranks. The modeling technique also relies on users' listening relation.

Our proposed algorithm can overcome the shortage caused by the cold start problem, because the algorithm does not rely on user's listening relations.

Recently a new research area called singing song recommendation was proposed by Mao et al. [13] in 2012. Shou et al. [14] proposed a competence based song recommendation framework to do song recommendation. Mao [15] then presented the singing song recommendation technique for the online social singing community. They focus on satisfying user's performance needs for singing. Our recommendation focuses on satisfying users' listening habits.

2.2. Graph ranking

Graph ranking algorithms focus on studying how to rank the graph vertices. The idea of the proposed methods [16–18] is mostly based on regularization theory. Zhou et al. [19] proposed a ranking algorithm which exploits the intrinsic manifold structure of the data. They built a weighted graph and performed random walk on the graph. Agarwal [20] developed an algorithmic framework for learning ranking functions on both directed and undirected graph data. They built the graph which acts as a regularizer to make sure that closely connected document will get a similar value.

Recently, many graph-based models are applied in multimedia and computer vision. They can be used as geometric image descriptors [21] to enhance image categorization. Besides, these methods can be used as image high-order potential descriptors of superpixels [22–25]. Further, graph-based descriptors can be used as general image aesthetic descriptors to improve image aesthetics ranking, photo-retargeting and cropping [26,27].

3. Framework overview

In this section, we introduce our recommendation framework. Our recommendation can be divided into two stages. Fig. 1 is a demonstration of the system framework.

In the first stage, we discover the preference relation from user's listening histories. We will measure whether a relation is reliable based on the two factors: the popularity of the relation and the agreement of the relation. The popularity of the relation measures whether the preference relation is accepted by many users. The agreement of the relation reflects whether there are many users opposing the relation.

In the second stage, we build a preference graph to model the preference orderings and design a ranking algorithm based on graphical inference to rank the songs for recommendation.

4. Preference relation

In this section, we present the techniques for discovering the preference relation between two tracks. We formally define the concept of preference relation in Section 4.1. Then we estimate the reliability of the preference relation in Section 4.2. Finally, we discuss the way to discover user's preference relation from users' ratings.

4.1. Preference relation

The user ratings in the music community indicate users' preferences on different tracks. The basic idea of our recommendation is as follows: if a user prefers a track, then he/she will also prefer the tracks which are more fair-sounding than the current track. Thus, we introduce the idea of *preference relation* between two tracks in order to represent whether a track is more fair-sounding than another track. We use $tr_i \rightarrow tr_j$ to represent that track tr_j is more fair-sounding than track tr_i.

For determining the preference relation between two tracks, we utilize the users' preference from the wisdom of the crowd. We judge the preference relation from users' preference ratings for each track. The basic idea is as follows. Suppose a group of users in the music community listens to two tracks tr_j and tr_i; they have rated the tracks according to their preferences. We can infer the preference of two songs from users' rating. For example, if user rates tr_j are higher than tr_i, then we know that his/her judgement for two tracks is $tr_i \rightarrow tr_j$. If there are more users preferring tr_j than tr_i, we can obtain a preference relation $tr_i \rightarrow tr_j$. First of all, we define the user's preference judgement for two tracks:

Definition 1 (*Preference judgement*). Preference judgement is the process of judging which track is more fair-sounding between two tracks tr_i and tr_j, where π_{ij} and π_{ji} represent the numbers of users that support $tr_i \rightarrow tr_j$ and $tr_j \rightarrow tr_i$ respectively.

Now we formally define the preference relation between two tracks.

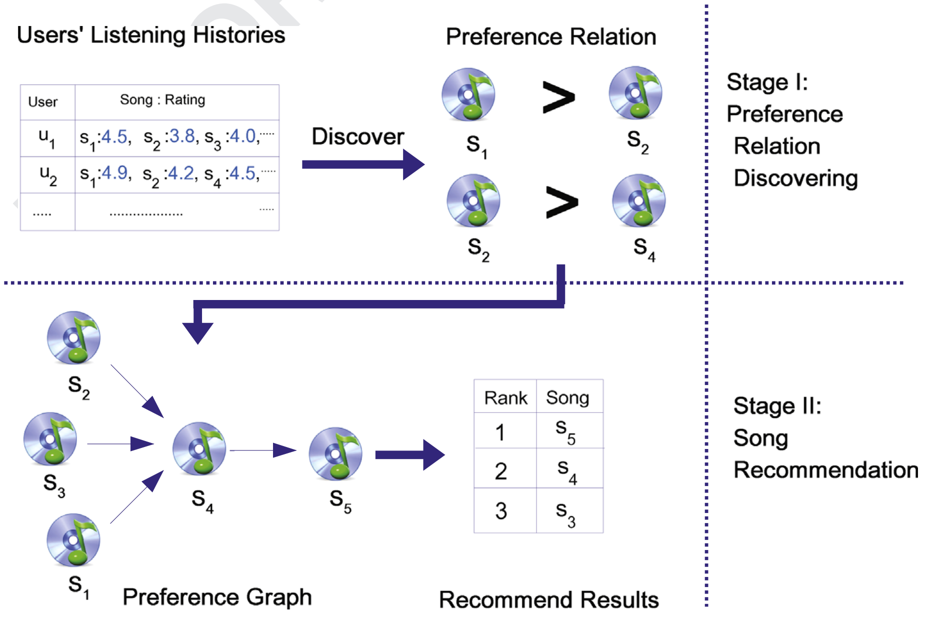
Users' Listening Histories
User
Song : Rating
u1
s1:4.5, s2:3.8, s3:4.0,.....
u2
s1:4.9, s2:4.2, s4:4.5,.....
Discover
Preference Relation
s1 > s2
s2 > s4
Stage I: Preference Relation Discovering
s2
s3
s1
s4
s5
Preference Graph
Rank Song
1 s5
2 s4
3 s3
Recommend Results
Stage II: Song Recommendation

Fig. 1. Overview of the song recommendation framework.

Definition 2 (*Preference relation*). The preference relation between two tracks tr_i and tr_j is the majority choice in preference judgment for $tr_i \to tr_j$ and $tr_j \to tr_i$.

4.2. Reliability of preference relation

As the users' choices are not always consistent on the preference relation, we should estimate whether the preference relation is reliable for music recommendation. We measure the reliability of the preference relation from two aspects: popularity and agreement.

4.2.1. Popularity

The popularity of the preference relation measures how many people agree with the relation $tr_i \to tr_j$. We denote the *Popularity* of the preference relation as $\texttt{pop}(tr_i \to tr_j)$. Intuitively, a preference relation that more people agree with will be more reliable than the one with less support. We define the popularity of the preference relation as follows:

Definition 3 (*Popularity of preference relation*). The popularity of a preference relation $tr_i \to tr_j$, denoted as $\texttt{pop}(tr_i \to tr_j)$, is the number of users who judge $tr_i \to tr_j$.

Because the track tr_i will also form preference relations with other tracks, we should measure the relative popularity of the preference relation in the tr_i related preference relations. The one with a large relative popularity would be more reliable to be used in recommendation. We define the relative popularity of a preference relation in a probabilistic form:

$$Pr_{pop}(tr_i \to tr_j) = \frac{\texttt{pop}(tr_i \to tr_j)}{\sum_{tr_k \in TR_\sigma} \texttt{pop}(tr_i \to tr_k)} \tag{1}$$

where TR_σ is a set of song tracks with each tr_k in TR_σ forming a preference relation $tr_i \to tr_k$.

4.2.2. Agreement

The popularity of the preference relation measures how many people agree with the relation. But it ignores the number of people who are opposite this preference relation. For example, if there are 10 users agree with the preference relation $tr_i \to tr_j$, however 8 users disagree with the judgment. This makes the preference relation unreliable in recommendation. So the *agreement* of the preference relation measures the total agreement of the users on the preference relations. We denote the agreement of the preference relation as $\texttt{agr}(tr_i \to tr_j)$. Now we formally define the concept of agreement for the preference relations.

Definition 4 (*Agreement of preference relation*). The agreement of the preference relation $tr_i \to tr_j$, denoted as $\texttt{agr}(tr_i \to tr_j)$, is the probability of getting the $tr_i \to tr_j$, given $tr_i \to tr_j$ and $tr_j \to tr_i$'s preference judgement results π_{ij} and π_{ji}.

Now we begin to estimate the agreement of the preference relation. According to the definition, the agreement of the preference relation can be represented as $Pr(tr_i \to tr_j | \pi_{ij}, \pi_{ji})$ Firstly, we apply Bayes's theorem:

$$Pr(tr_i \to tr_j | \pi_{ij}, \pi_{ji}) = \frac{Pr(\pi_{ij}, \pi_{ji} | tr_i \to tr_j) P(tr_i \to tr_j)}{P(\pi_{ij}, \pi_{ji})} \tag{2}$$

We also decompose the agreement of the preference relation $tr_j \to tr_i$

$$Pr(tr_j \to tr_i | \pi_{ji}, \pi_{ij}) = \frac{Pr(\pi_{ji}, \pi_{ij} | tr_j \to tr_i) P(tr_j \to tr_i)}{P(\pi_{ji}, \pi_{ij})} \tag{3}$$

We do not have prior knowledge on which preference relation is more possible, we assume that they have the same probability to be preferred. So we combine (2) and (3), and get

$$\frac{Pr(tr_i \to tr_j | \pi_{ij}, \pi_{ji})}{Pr(tr_j \to tr_i | \pi_{ji}, \pi_{ij})} = \frac{Pr(\pi_{ij}, \pi_{ji} | tr_i \to tr_j)}{Pr(\pi_{ji}, \pi_{ij} | tr_j \to tr_i)} \tag{4}$$

We can assume that the sum of the agreements of two opposite preference relations equals to 1, so we can transform 4 into

$$Pr(tr_i \to tr_j | \pi_{ij}, \pi_{ji}) = \frac{Pr(\pi_{ij}, \pi_{ji} | tr_i \to tr_j)}{Pr(\pi_{ij}, \pi_{ji} | tr_i \to tr_j) + Pr(\pi_{ji}, \pi_{ij} | tr_j \to tr_i)} \tag{5}$$

We can see that the agreement of the preference relation can be represented using the probability of getting the preference judgement results, knowing the preference ordering. Because the user can only choose one of the opposite preference relations in judgment, we consider this as a Bernoulli trial with q as the judgment accuracy. The probability of observing the judgment results given the preference ordering can be estimated as

$$Pr(\pi_{ij}, \pi_{ji} | tr_i \to tr_j) = C^{\pi_{ij}}_{\pi_{ij} + \pi_{ji}} q^{\pi_{ij}} (1-q)^{\pi_{ji}} \tag{6}$$

$$P(\pi_{ji}, \pi_{ij} | tr_j \to tr_i) = C^{\pi_{ji}}_{\pi_{ij} + \pi_{ji}} q^{\pi_{ji}} (1-q)^{\pi_{ij}} \tag{7}$$

4.2.3. Reliability

Knowing the popularity and agreement of the preference relation, we can estimate the reliability of the preference relation in recommendation. We consider those preference relations with both high relative popularity and agreement as the one with high reliability. We define the reliability of a preference relation as follows.

Definition 5 (*Reliability of preference relation*). The reliability of the preference relation $tr_i \to tr_j$, denotes as $\texttt{Reli}(tr_i \to tr_j)$, is the probability that both $tr_i \to tr_j$'s relative popularity and agreement are high.

As the definition,

$$\texttt{Reli}(tr_i \to tr_j) = Pr(\texttt{pop}(tr_i \to tr_j), \texttt{agr}(tr_i \to tr_j)) \tag{8}$$

Because the popularity and the agreement of the preference relation to be high are two independent events, we can estimate the reliability of the preference relation as

$$\texttt{Reli}(tr_i \to tr_j) = Pr_{pop}(tr_i \to tr_j) * \texttt{agr}(tr_i \to tr_j) \tag{9}$$

4.3. Preference relation discovery

To discover the preference relation from the online music community, we first deduce users' judgments towards different tracks. Suppose we have a user who has listened to two tracks tr_i and tr_j. They will rate the preference to these two tracks. We deduce users' preference as follows. First of all, user u listens to track tr_i and tr_j and rates the two tracks with scores r_i and r_j respectively. If $r_i < r_j$ means u likes tr_j better than tr_i and vice versa. This indicates that u's judgment for the two tracks is $tr_i \rightarrow tr_j$.

Now we introduce how to discover song preference relations from users' listening history. First of all, we discover those track pairs which are rated by 10 users. Then we deduce the users' judgment for each track pair as follows. For each track pair tr_i and tr_j, we count the judgment for $tr_i \rightarrow tr_j$ and $tr_j \rightarrow tr_i$ respectively. We consider the one in $tr_i \rightarrow tr_j$ and $tr_j \rightarrow tr_i$ with more users' supports as the preference relation between track tr_i and tr_j. Now we can calculate the reliability of the preference relation using Eq. (9).

5. Graph based quality model

In this section, we introduce our graph based quality model to do music recommendation based on the above discovered preference relations. The quality model is used to integrate all the preference relations so as to be used for music recommendation. The basic idea is to measure each track's probability of being preferred by the user. We call this probability as preference probability. We use the preference probability to rank the tracks for recommendation. For example, if we know tr_i have a larger preference probability than tr_j, we should prefer tr_i as the recommendation for the user. In the following part, we will introduce our recommendation algorithm. We introduce the basic component of our recommendation algorithm in Section 5.1 which describes how to do graphical inference in our quality model. In Section 5.2, we will introduce our song ranking algorithm which is actually an optimization problem upon the quality model.

5.1. Graphical inference

In this section, we will introduce how to estimate the preference probability of each track from the preference relations through graphical inference in our quality model (we call it as preference graph). First of all, we define the preference probability for each track. The preference probability of track tr_j for user u_k is the probability that u_k likes to listen track tr_j

$$pr_j^k = Pr\{u_k \text{ likes } tr_j\}$$

To estimate each track's preference probability, we first model our quality model as a preference graph. Preference graph is a direct graph, with each track as the vertex, we denote the edge weight from track tr_i to track tr_j as w_{ij}. We define a preference graph as follows:

Definition 6 (*Preference graph*). A Preference Graph G is an ordinary directed weighted graph where the vertex set Tr are tracks, each edge from track tr_i to tr_j represents a preference relation with w_{ij} as the weight.

We set the edge weight w_{ij} as the reliability of the preference relations. Now we start to introduce how to perform graphical inference to estimate the preference probability of each track. We have estimated the preference probability of track tr_j. Suppose we know the preference probability of its in-degree tracks, I will introduce how we perform graphical inference to estimate the preference probability. We represent the event that u_k likes to listen tr_j as $L(tr_j^k)$. Based on the graph theory, we use the sum and product rule of probability and get

$$Pr(L(tr_j^k)) = \sum_{tr_i^k \in TR_{in}^j} [Pr(L(tr_j^k)|L(tr_i^k)) * Pr(L(tr_i^k))] \tag{10}$$

Knowing the preference probability of tr_j's in-degree tracks, we only need to estimate $Pr(L(tr_j^k)|L(tr_i^k))$, this is the probability that user u_k likes to listen tr_j, knowing the fact that u_k likes to listen tr_i which is the reliability of the preference ordering.

But the problem is the user only listens to very few songs in the graph, how to do graphical inference if we do not know the preference probability of one track's in-degree tracks. In the next part, we will introduce a regularization framework which integrates the probability inference algorithm into preference probability estimation. The advantage of the framework is that it does not need to know the actual preference degree of each track to do recommendation.

5.2. Regularization framework

In this part, we introduce a regularization framework to predict the preference probability of each track in the preference graph by implicitly performing the graphical inference. The basic idea is a kind of information spreading. Starting from user's previous listened tracks, we spread the information to other tracks in the preference graph using the graphical inference described in Section 5.1.

We denote the preference probability of each vertex as $\mathbf{f} = [f_1, f_2, \ldots, f_{|Tr|}]$, where $|Tr|$ is the number of vertices in the preference graph. We construct the weight matrix $\mathbf{W}$ which is a $|Tr| \times |Tr|$ matrix, where W_{ij} equals to the weight from track tr_i to tr_j, that is $W_{ij} = w_{ij}$. We let $\mathbf{D}$ as a diagonal matrix with its (i,i)-element equal to the sum of the i-th row of $\mathbf{W}$. We define a vector $\mathbf{y} = [y_1, y_2, \ldots, y_{|Tr|}]$. If the user had listened to tr_i, then $y_i = 1$, otherwise 0. We develop a regularization framework for calculating the preference probability of each vertex ($\mathbf{f}$ vector). First of all we define a cost function associated with $\mathbf{f}$ as follows:

$$Q(\mathbf{f}) = \frac{1}{2} \sum_{i,j=1}^{n} W_{ij} \left\| \frac{f_i}{\sqrt{D_{ii}}} - \frac{f_j}{\sqrt{D_{jj}}} \right\|^2 + \mu \sum_{i=1}^{n} \| f_i - y_i \|^2 \tag{11}$$

where μ is the regularization parameter. Our objective is to minimize the cost function:

$$\mathbf{f}^* = \underset{\mathbf{f} \in \mathcal{F}}{\operatorname{argmin}}\, Q(\mathbf{f}) \tag{12}$$

The first term on the right-hand side in Eq. (11) is the smoothness constraint, which means that the result of the information spreading in the preference graph should satisfy that the information of nearby points should not change too much. The second term is the fitting constraint, which means the final preference probability should not change too much comparing with the initial label $\mathbf{y}$. These two constraints are contradictory which makes sure that the cost function can be minimized. The trade-off between these two competing constraints is captured by a positive parameter μ. Now we deduce the expression for $\mathbf{f}$. First of all, we differentiating $Q(\mathbf{f})$ with respect to $\mathbf{f}$, we have

$$\left.\frac{\partial Q}{\partial \mathbf{f}}\right|_{\mathbf{f}=\mathbf{f}^*} = \mathbf{f}^* - T\mathbf{f}^* + \mu(\mathbf{f}^* - \mathbf{y}) = 0, \tag{13}$$

where $T = D^{-1/2}WD^{-1/2}$, and we transform the above equation into

$$\mathbf{f}^* - \frac{1}{1+\mu}T\mathbf{f}^* - \frac{\mu}{1+\mu}\mathbf{y} = 0. \tag{14}$$

We use α and β to represent

$$\alpha = \frac{1}{1+\mu} \quad \text{and} \quad \beta = \frac{\mu}{1+\mu}. \tag{15}$$

Because $\alpha+\beta=1$. Then

$$(I-\alpha T)\mathbf{f}^* = \beta\mathbf{y}. \tag{16}$$

Since $I-\alpha S$ is invertible, we have

$$\mathbf{f}^* = \beta(I-\alpha T)^{-1}\mathbf{y}. \tag{17}$$

Now knowing W matrix which is the edge weight matrix and users' previous listening tracks $\mathbf{y}$, we can estimate the preference probability $\mathbf{f}^*$ for each song track.

6. Experimental results

6.1. Datasets

We collect data from the Last.fm which is a very popular website for listening to music. We download 1 million user's listening histories as our experimental data.

For the preference relation discovery, we crawl those song pairs which have been listened to by at least 10 users. We model these song pairs into preference relations. Finally, we totally collect 9845 songs which form 212 234 preference relations for song recommendation.

In order to evaluate the proposed framework, we filter our 5000 users and their listening histories. The number of songs listened by each user is 958. We randomly select 30% listening history of each user as the testing set and others as the train set. We ensure there is no overlapping between each user's training and testing set.

6.2. Evaluation metric

We use sensitive evaluation metrics such as F1, Normalized Discount Cumulative Gain (NDCG), and Mean Average Precision (MAP) to measure the recommendation performance.

Precision@n and recall@n reflect the precision and recall value at the specific position in the ranking list. Precision@n is defined as the number of correctly recommended songs divided by the total number of songs recommended at position n, while recall@n is defined as the correctly recommended songs divided by the total number of songs which should be recommended in the test set. F1 is the harmonic mean of Precision and Recall. It can be calculated as

$$F1 = \frac{2 * Precision@n * Recall@n}{Precision@n + Recall@n} \tag{18}$$

We use mean average precision (MAP) [28] to compare the ranking accuracy measured from the whole rank list. For each user, it first calculates the precision at the position of each relevant song in the recommended list. Then these precision values are reduced by their rank positions so that top ranked relevant documents contribute more to the final score. The sum of the precision values is averaged by the total number of correctly recommended items. Now we get the average precision (AP). MAP is calculated by averaging the AP value of each query:

$$MAP = \frac{1}{|Q|}\sum_{q\in Q}\frac{\sum_{n=1}^{N}(Precision@n * rel(n))}{R_q} \tag{19}$$

where Q is the test query set, R_q is q's relevant document, $rel(n)$ is a binary function on the relevance of a given rank, and N is the number of retrieved documents.

We use NGCG@n [29] to measure the accuracy of the rank order at a specific position of the ranking result. It is calculated as follows:

$$N\,DCG(n) = \frac{1}{Z_n}\sum_{j=1}^{n}\frac{2^{r(j)}-1}{\log_2(j+1)} \tag{20}$$

where j is the predicted rank position, $r(j)$ is the rating value of j-th document in the ground-truth rank list, Z_n is the normalization factor which is the discounted cumulative gain in the n-th position of the ground-truth rank list. We consider $r(j)$ as 1 if the song is sung by the user, otherwise 0.

6.3. Baselines

We compare our graph based algorithm with two state-of-art baselines. We call our algorithms as Preference Graph (PG) based method.

The first baseline is the user-based collaborative filtering (CF) method. We build a user-song matrix using users' rating histories. We first predict the rating for those unrated songs and then rank the songs based on the predicted ratings.

The second baseline is User-Item Graph (UIG) based method. We construct a graph with user and song as vertex, the edge represents the listening relation.

6.4. Experimental results

6.4.1. Baseline comparison

We use the metric introduced in Section 6.2 to compare the performance of different algorithms. We set the α used in PG and UIF as 0.95 and the human judgment accuracy as 0.78.

Fig. 3 is the result measured from NDCG@n. We can find that our algorithm PG outperforms the two baselines by an average of 150% and 20%. This demonstrates the effectiveness of our proposed framework. Fig. 2 is the F1 result for the three algorithms. PG outperforms CF and UIG by an average of 80% and 10%, respectively. The reason why PG outperforms CF and UIG is that PG considers the precise fair-sounding relations between different songs. These preference orderings are discovered from user's actual ratings. The recommendation based on these rules is reliable.

Table 1 is the result measured from MAP which is an overall recommendation accuracy measure. We can find that our algorithm PG is better than CF and UIG by 87% and

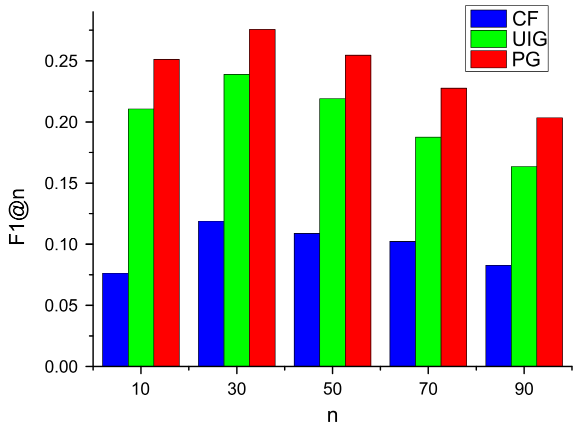

Fig. 2. Performance measured from F1@n.

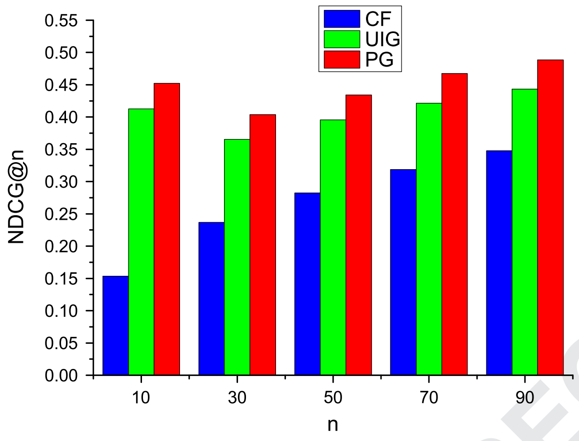

Fig. 3. Performance measured from NDCG@n.

Table 1
Ranking accuracy in terms of MAP.

Algorithm	CF	UIG	PG
MAP value	0.1389	0.2253	0.2608

15%, respectively. We can see that the improvement is not so high comparing with the other metrics. It shows that the PG is good at recommending top ranked songs. For an overall viewpoint, UIG also can acquire a good result.

6.4.2. Cold start problem

We also test our proposed framework in a sparse dataset where each user only has 5–10 listening songs in the history. These users are called cold start user. The traditional algorithms will not perform well due to the lack of the listening relation. The total user in the dataset is 8200.

Fig. 4(a) and (b) shows the F1 and NDCG@n results of the three algorithms. We find that our algorithm performs much better than the baseline in the cold start dataset. This proves the effectiveness of PG in overcoming the cold start problem in recommendation system. We can see from the result that UIG's performance drops quite a lot. This is because the graph building process in UIG relies on the user's listening relations. In the sparse dataset, the lack of the listening relations causes many isolate vertices in the graph, it makes the recommendation algorithm fails to perform well. Our recommendation framework will not be affected a lot by the data sparse problem. Our pre-built preference graph can overcome the data sparse problem in recommendation.

6.4.3. Conclusion and future work

In this paper, we propose a music recommendation framework based on graph based quality model to make fine-grained music recommendation. We first discover the recommendation cues, which we called user's preference relations from the users' ratings. Then we model them using the quality model and propose a regularization framework to calculate the recommendation probability of songs. Our experiments show that the proposed framework is superior to two traditional algorithms, especially in solving the cold start problem in recommendation.

For future work, we will model other factors such as friend relation, group information, tag information and so on into the preference graph to study whether they can help to improve the performance.

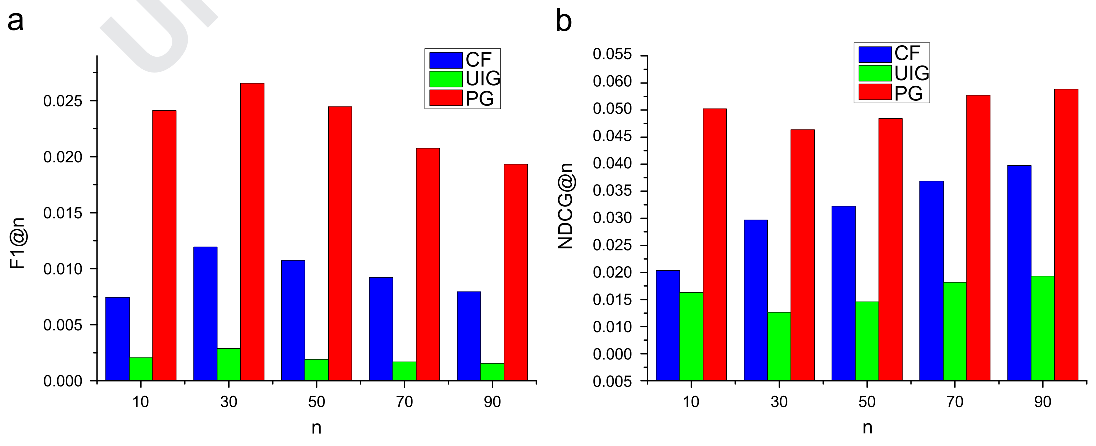

Fig. 4. Performance comparison for cold start user (a) F1@n. (b) NDCG@n.

References

[1] B. Logan, Music recommendation from song sets, in: Proceedings of ISMIR, 2004, pp. 425–428.

[2] P. Cano, Content-based music audio recommendation, in: ACM Multimedia, 2005.

Q3 [3] R. Cai, C. Zhang, L. Zhang, W.-Y. Ma, Scalable music recommendation by search, in: R. Lienhart, A.R. Prasad, A. Hanjalic, S. Choi, B.P. Bailey, N. Sebe (Eds.), ACM Multimedia, ACM, 2007, pp. 1065–1074.

[4] S. Rho, B. Jun Han, E. Hwang, SVR-based music mood classification and context-based music recommendation, in: W. Gao, Y. Rui, A. Hanjalic, C. Xu, E.G. Steinbach, A. El-Saddik, M.X. Zhou (Eds.), ACM Multimedia, ACM, 2009. Q4

[5] K. Yoshii, M. Goto, K. Komatani, T. Ogata, H.G. Okuno, Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences, in: ISMIR, 2006, pp. 296–301.

[6] Q. Li, S.-H. Myaeng, B.M. Kim, A probabilistic music recommender considering user opinions and audio features, Inf. Process. Manag. 43 (2) (2007) 473–487.

[7] M. Tiemann, S. Pauws, Towards ensemble learning for hybrid music recommendation, in: J.A. Konstan, J. Riedl, B. Smyth (Eds.), RecSys, ACM, 2007, pp. 177–178.

[8] K. Yoshii, M. Goto, Continuous pLSI and smoothing techniques for hybrid music recommendation, in: K. Hirata, G. Tzanetakis, K. Yoshii (Eds.), ISMIR, International Society for Music Information Retrieval, 2009, pp. 339–344.

[9] Ò Celma, X. Serra, FOAFing the music: bridging the semantic gap in music recommendation, J. Web Sem. 6 (4) (2008) 250–256.

[10] D. Goldberg, D.A. Nichols, B.M. Oki, D.B. Terry, Using collaborative filtering to weave an information tapestry, Commun. ACM 35 (12) (1992) 61–70.

[11] J.S. Breese, D. Heckerman, C.M. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, CoRR abs/1301.7363.

[12] B.M. Sarwar, G. Karypis, J.A. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: WWW, 2001, pp. 285–295.

[13] K. Mao, X. Luo, K. Chen, G. Chen, L. Shou, myDJ: recommending Karaoke songs from one's own voice, in: SIGIR, 2012, p. 1009.

[14] L. Shou, K. Mao, X. Luo, K. Chen, G. Chen, T. Hu, Competence-based song recommendation, in: SIGIR, 2013, pp. 423–432.

[15] K. Mao, J. Fan, L. Shou, G. Chen, M. Kankanhalli, Competence-based song recommendation, in: ACM Multimedia, 2014.

[16] M. Belkin, I. Matveeva, P. Niyogi, Regularization and Semi-supervised Learning on Large Graphs, vol. 3120, Springer, 2004, 624–638.

[17] M. Belkin, P. Niyogi, Semi-supervised learning on Riemannian manifolds, Mach. Learn. 56 (1–3) (2004) 209–239.

[18] D. Zhou, J. Huang, B. Schölkopf, Learning from labeled and unlabeled data on a directed graph, in: ICML, vol. 119, 2005, pp. 1036–1043.

[19] D. Zhou, J. Weston, A. Gretton, O. Bousquet, B. Schölkopf, Ranking on data manifolds, in: Advances in Neural Information Processing Systems 16, MIT Press, 2004.

[20] S. Agarwal, Ranking on graph data, in: W.W. Cohen, A. Moore (Eds.), ICML, vol. 148, ACM, 2006, pp. 25–32.

[21] L. Zhang, Y. Han, Y. Yang, M. Song, S. Yan, Q. Tian, Discovering discriminative graphlets for aerial image categories recognition, IEEE Trans. Image Process. 22 (12) (2013) 5071–5084.

[22] L. Zhang, Y. Yang, Y. Gao, C. Wang, Y. Yu, X. Li, A probabilistic associative model for segmenting weakly-supervised images, IEEE Trans. Image Process. (2014). Q2

[23] L. Zhang, Y. Gao, R. Ji, L. Ke, J. Shen, Representative discovery of structure cues for weakly-supervised image segmentation, IEEE Trans. Multimed. 16 (2) (2014) 470–490.

[24] L. Zhang, M. Song, Z. Liu, X. Liu, J. Bu, C. Chen, Probabilistic graphlet cut: exploring spatial structure cue for weakly supervised image segmentation, in: IEEE Computer Vision and Pattern Recognition (CVPR), 2013.

[25] L. Zhang, M. Song, X. Liu, J. Bu, C. Chen, Fast multi-view segment graph kernel for object classification, Signal Process. (2013) 1597–1607.

[26] L. Zhang, M. Song, Q. Zhao, X. Liu, J. Bu, C. Chen, Probabilistic graphlet transfer for photo cropping, IEEE Trans. Image Process. 21 (5) (2013) 2887–2897.

[27] L. Zhang, Y. Gao, R. Zimmermann, Q. Tian, X. Li, Fusion of multichannel local and global structural cues for photo aesthetics evaluation, in: IEEE Trans. Image Process. 23 (3) (2014) 1419–1429.

[28] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley, 1999.

[29] K. Järvelin, J. Kekäläinen, Ir evaluation methods for retrieving highly relevant documents, in: SIGIR, 2000, pp. 41–48.