Addressing Cold Start for Next-song Recommendation

Szu-Yu Chou[1,2], Yi-Hsuan Yang[2], Jyh-Shing Roger Jang[1], Yu-Ching Lin
[1]Graduate Institute of Networking and Multimedia, National Taiwan University, Taipei, Taiwan
[2]Research Center for IT innovation, Academia Sinica, Taipei, Taiwan
{fearofchou,yang}@citi.sinica.edu.tw,
jang@csie.ntu.edu.tw, aaronlin@kkbox.com

ABSTRACT

The cold start problem arises in various recommendation applications. In this paper, we propose a tensor factorization-based algorithm that exploits content features extracted from music audio to deal with the cold start problem for the emerging application next-song recommendation. Specifically, the new algorithm learns sequential behavior to predict the next song that a user would be interested in based on the last song the user just listened to. A unique characteristic of the algorithm is that it learns and updates the mapping between the audio feature space and the item latent space each time during the iterations of the factorization process. This way, the content features can be better exploited in forming the latent features for both users and items, leading to more effective solutions for cold-start recommendation. Evaluation on a large-scale music recommendation dataset shows that the recommendation result of the proposed algorithm exhibits not only higher accuracy but also better novelty and diversity, suggesting its applicability in helping a user explore new items in next-item recommendation. Our implementation is available at https://github.com/fearofchou/ALMM.

Keywords

Next-song recommendation, content-based recommendation, matrix factorization, real-life setting, context-aware system

1. INTRODUCTION

Music recommendation is extensively used by online music streaming service providers, such as Spotify and Pandora.[1] The goal of a recommendation algorithm is to model the preference of users from observed user-item associations (either explicit or implicit feedback), and then use the model to predict the items a user may like but is not aware before.

The user preference also can be modeled with additional context information [8,10] and content feature [2,6,9]. Music recommendation can be *contextualized* in a number of ways, for example by taking into account the time or location of music listening, the ongoing activity of the user, or the underlying emotional state of the user, amongst others. In this paper, we are interested in the scenario where we are given the very last song a user just listened to, and are tasked to predict the next song the user may like to have. The most recently played song may reflect the user's current preference and thereby contribute to the modeling of the listening context. In consequence, such a *next-item recommendation* may be useful in real-world applications [8,10].

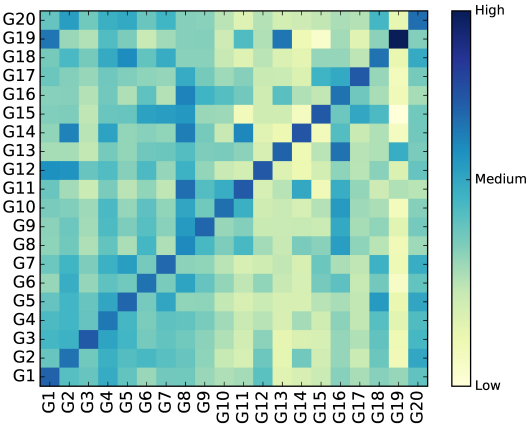

Figure 1: Probability of song-to-song transition between music of different genres, observed from our dataset.

As an illustration, we show in Figure 1 the transition probability of songs of different musical genres in the dataset studied in this paper. The probability is estimated by the Markov chain (MC) method [8], disregarding transitions between a song and itself. The high values along the diagonal shows that the next song has a high probability of being of the same genre as the last song to which the user just listened, which may not be surprising as many users continuously listen to the same artist or album. We also see that the matrix is not a symmetric one, and that there are a few high off-diagonal values, suggesting potentially interesting sequential patterns to be employed in next-item recommendation.

Sequential behavior prediction based on MC is one of the most popular methods for capturing transition preferences. On the other hand, matrix factorization (MF) has been successfully used in a variety of recommendation problems. To exploit the advantages of the two methods, Rendle *et al.* [8] combined MC and MF to improve next-item recommendation.

[1] http://www.spotify.com, http://www.pandora.com

RecSys '16, September 15–19, 2016, Boston, MA, USA.

DOI: http://dx.doi.org/10.1145/2959100.2959156

The approach counts the number of transitions between pairs of items for each user, and then learns latent feature representations for the users and items from the resulting {user, item, last-time item} triplets. This leads to *user, item, and last-time item latent vectors*, whose inner product can be used to perform next-item recommendation.

For the specific application of next-song recommendation, it is important to deal with the so-called *cold-start* problem [6] associated with newly released or less popular songs, which are commonly seen in real-world music streaming services. It is well-known that MF-based methods, or other collaborative filtering (CF)-based methods in general, cannot perform well for items with sparse association with users in the training data. Due to the cold-start problem, the recommendation result may lack diversity and novelty, which are both important for user satisfaction. While CB methods may remedy this issue by exploiting item-item association in the audio feature space, little effort has been made to develop content-based algorithms for next-item recommendation.

The specific goal of this paper is to address cold start next-song recommendation. Following Rendle *et al.*'s approach [8], the new algorithm additionally learns a linear transformation matrix to learn a mapping from an audio feature space to the item latent feature space, each time as we update all the latent features in the tensor factorization process. In other words, the mapping between the content features and song latent vectors is constructed in an interactive way *during*, instead of *after* (e.g. as a prior work [6] did), the factorization process. This way, the coupling between content features and item latent vectors is made stronger. Moreover, the content features would affect not only the learning of the item latent and last-time item latent features, but also the user latent features. In our experiments, the proposed method leads to better accuracy as compared with two competing methods in both warm-start and cold-start settings. Moreover, its recommendation features higher novelty, diversity and lower popularity, which are desirable to user satisfaction.

2. NEXT-SONG RECOMMENDATION

2.1 Formalization

The goal is to recommend songs to a specific user, given knowledge of the song the user just listened to. Let $U = (u_1, u_2, \ldots, u_{|U|})$ be a set of users and $S = (s_1, s_2, \ldots, s_{|S|})$ be a set of songs. According to the listening timestamp, we have the listening sequence $L^u = (L^u_1, L^u_2, \ldots, L^u_t)$ for each user u. The listening sequences of all users is collectively $L = (L^{u_1}, L^{u_2}, \ldots, L^{u_{|U|}})$. We can convert the listening sequences to a transition matrix between different songs for each user by counting the number of adjacent song pairs. To mine sequential patterns, we remove the transitions between a song and itself, and require the time interval between adjacent songs to fall within half an hour to be considered as a valid pair. The resulting tensor $P \in \mathbb{R}^{|U|\times|S|\times|S|}$ for the transition preference for each user is defined as:

$$P^u_{i,j} = \sum_{u\in U} \sum_{n\in L_{t-1}} |\{(L^u_n, L^u_{n+1}) : i \in L^u_n \wedge j \in L^u_{n+1}\}| . \quad (1)$$

Given P, a model can be trained to learn personal transition preference and to recommend a number of songs to a user based on the very last song the user just listened to. Following Hu *et al.* [3], we transform the counts $P^u_{i,j}$ into *confidence values* $C^u_{i,j}$ by the relation $C^u_{i,j} = 1 + \alpha \log(1 + P^u_{i,j}/\varepsilon)$, where α and ε are user-defined parameters, which are both set to 1 in this work.

2.2 Pairwise Factorization (PF)

We present the pairwise factorization (PF) model for personalized next-song recommendation. Following [7], the model factorizes all the features by using pairwise interaction. The objective function is defined as:

$$\min_{\boldsymbol{U}*,\boldsymbol{X}*,\boldsymbol{Y}*} \sum_{(u,i,j)} (C^u_{i,j} - \boldsymbol{U}_u^T \boldsymbol{X}_i - \boldsymbol{U}_u^T \boldsymbol{Y}_j - \boldsymbol{X}_i^T \boldsymbol{Y}_j)^2 + \lambda_U \|\boldsymbol{U}_u\|^2 + \lambda_X \|\boldsymbol{X}_i\|^2 + \lambda_Y \|\boldsymbol{Y}_j\|^2 \quad (2)$$

where $\boldsymbol{U_u} \in \mathbb{R}^k$ is a user latent vector, $\boldsymbol{X_i} \in \mathbb{R}^k$ a last-time item latent vector, $\boldsymbol{Y_j} \in \mathbb{R}^k$ a item latent vector, and k the latent dimension. The updating rules for minimizing (2) can be obtained through alternating least squares:

$$\boldsymbol{U}_u = \Big(\sum_{i,j\in \boldsymbol{C}} ((\boldsymbol{XY})_{ij} (\boldsymbol{XY})^T_{ij}) + \lambda_U \Big)^{-1} \Big(\sum_{i,j\in \boldsymbol{C}} (C^u_{i,j} - \boldsymbol{Y}_j^T \boldsymbol{X}_i)(\boldsymbol{XY})_{ij} \Big) \quad (3)$$

$$\boldsymbol{X}_i = \Big(\sum_{u,j\in \boldsymbol{C}} ((\boldsymbol{UY})_{uj} (\boldsymbol{UY})^T_{uj}) + \lambda_X \Big)^{-1} \Big(\sum_{u,j\in \boldsymbol{C}} (C^u_{i,j} - \boldsymbol{Y}_j^T \boldsymbol{U}_u)(\boldsymbol{UY})_{uj} \Big) \quad (4)$$

$$\boldsymbol{Y}_j = \Big(\sum_{u,i\in \boldsymbol{C}} ((\boldsymbol{UX})_{ui} (\boldsymbol{UX})^T_{ui}) + \lambda_Y \Big)^{-1} \Big(\sum_{u,i\in \boldsymbol{C}} (C^u_{i,j} - \boldsymbol{X}_i^T \boldsymbol{U}_u)(\boldsymbol{UX})_{ui} \Big) \quad (5)$$

where we use the shorthands $(\boldsymbol{UX})_{ui}$, $(\boldsymbol{UY})_{uj}$ and $(\boldsymbol{XY})_{ij}$ to denote $\boldsymbol{U}_u + \boldsymbol{X}_i$, $\boldsymbol{U}_u + \boldsymbol{Y}_j$ and $\boldsymbol{X}_i + \boldsymbol{Y}_j$ respectively.

2.3 Content-based Next-song Recommendation

This section elaborates two baseline CB methods and the proposed method for content-based next-song recommendation. Given the audio features $\boldsymbol{A}_i \in \mathbb{R}^m$ for song i, the common goal of these methods is to learn a mapping matrix $\boldsymbol{\Psi} \in \mathbb{R}^{k\times m}$ between the audio space feature space and the item latent space. We need two mappings, $\boldsymbol{\Psi}^X$ and $\boldsymbol{\Psi}^Y$, for the last-time item and item latent features, respectively.

2.3.1 Baseline Content-based approaches

Forbes [2]: The item latent vector is learned directly as a linear transformation of the content features in tensor factorization. The objective function is defined as:

$$\min_{\boldsymbol{U}*,\boldsymbol{\Psi}^X*,\boldsymbol{\Psi}^Y*} \sum_{(u,i,j)} \Big(C^u_{i,j} - \boldsymbol{U}_u^T \boldsymbol{\Psi}^X \boldsymbol{A}_i - \boldsymbol{U}_u^T \boldsymbol{\Psi}^Y \boldsymbol{A}_j - (\boldsymbol{\Psi}^X \boldsymbol{A}_i)^T (\boldsymbol{\Psi}^Y \boldsymbol{A}_j) \Big)^2 + \lambda_U \|\boldsymbol{U}_u\|^2 + \lambda_{\Psi^X} \|\boldsymbol{\Psi}^X\|^2 + \lambda_{\Psi^Y} \|\boldsymbol{\Psi}^Y\|^2 . \quad (6)$$

In other words, the content features are treated as constraints in forming the item-related latent features $\boldsymbol{\Psi}^X \boldsymbol{A}_i$ and $\boldsymbol{\Psi}^Y \boldsymbol{A}_j$. For each training entry, we compute the corresponding prediction error $e^u_{i,j}$:

$$e^u_{i,j} = C^u_{i,j} - \boldsymbol{U}_u^T \boldsymbol{\Psi}^X \boldsymbol{A}_i - \boldsymbol{U}_u^T \boldsymbol{\Psi}^Y \boldsymbol{A}_j - (\boldsymbol{\Psi}^X \boldsymbol{A}_i)^T (\boldsymbol{\Psi}^Y \boldsymbol{A}_j) . \quad (7)$$

Applying an alternating gradient descent algorithm, the following updating rules for $\boldsymbol{U}_u$, $\boldsymbol{\Psi}^X$ and $\boldsymbol{\Psi}^Y$ are obtained:

$$\boldsymbol{U}_u \leftarrow \boldsymbol{U}_u + \mu \Big(\sum_{(u,i,j)\in \boldsymbol{C}} e^u_{i,j} (\boldsymbol{\Psi}^X A_i + \boldsymbol{\Psi}^Y A_j) - \lambda_U \boldsymbol{U}_u \Big) , \quad (8)$$

$$\boldsymbol{\Psi}^X \leftarrow \boldsymbol{\Psi}^X + \mu \Big(\sum_{(u,i,j)\in \boldsymbol{C}} e^u_{i,j} (\boldsymbol{U}_u A_i^T + \boldsymbol{\Psi}^Y A_j A_i^T) - \lambda_{\Psi^X} \boldsymbol{\Psi}^X \Big) , \quad (9)$$

$$\boldsymbol{\Psi}^Y \leftarrow \boldsymbol{\Psi}^Y + \mu \Big(\sum_{(u,i,j)\in \boldsymbol{C}} e^u_{i,j} (\boldsymbol{U}_u A_j^T + \boldsymbol{\Psi}^X A_i A_j^T) - \lambda_{\Psi^Y} \boldsymbol{\Psi}^Y \Big) , \quad (10)$$

where μ is learning rate.

Table 1: The statistics of the training, validation and two test sets in our experiments

Data sets	#users	#songs	#entries
Train	26k	74k	3,004k
Validation	22k	35k	429k
Warm start (WS)	24k	47k	858k
Cold start (CS)	13k	10k	92k

Oord [6]: Unlike Forbes, here the mapping matrices $\boldsymbol{\Psi}^X$ and $\boldsymbol{\Psi}^Y$ are learned after the latent vectors have been obtained from tensor factorization. The objective function maps all the estimated song latent vectors from audio features, by minimizing the mean squared error.

$$\min_{\Psi^X} \sum_i (\boldsymbol{X}_i - \hat{\boldsymbol{X}}_i)^2 \,, \tag{11}$$

$$\min_{\Psi^Y} \sum_j (\boldsymbol{Y}_j - \hat{\boldsymbol{Y}}_j)^2 \,, \tag{12}$$

where $\hat{\boldsymbol{X}}_i \equiv \boldsymbol{\Psi}^X \boldsymbol{A}_i$ and $\hat{\boldsymbol{Y}}_j \equiv \boldsymbol{\Psi}^Y \boldsymbol{A}_j$. For fair comparison, the two baseline models Forbes and Oord are learned with similar settings.

2.3.2 *Adaptive linear mapping model (ALMM)*

The two baseline methods have different limitations. The Forbes method [2] might not obtain good latent features, as the constraints imposed by the content features might be too strong. On the other hand, the Oord method [6] is more flexible, but as the mapping is learned after tensor decomposition, the inner product between the user latent vectors $\boldsymbol{U}_u$ and the mapped item latent vector $\hat{\boldsymbol{Y}}_j$ might not be able to accurately predict user preference.

To deal with this issues, we propose the *adaptive linear mapping model* (ALMM) method, which learns the mapped latent vectors during, instead of after, iterations of the factorization process. Specifically, the proposed method follows the PF method and optimizes the user and item latent features using (3)–(5), but at the same time updates the mapping matrices such that the inner product among $\boldsymbol{U}_u$ and the mapped last-time item and item latent vector $\hat{\boldsymbol{X}}_i$ and $\hat{\boldsymbol{Y}}_j$ predicts user transition preference. The content feature and all the latent vectors can be therefore coupled strongly.

The above steps are summarized in Algorithm 1. The updating rules for $\boldsymbol{\Psi}^X$ and $\boldsymbol{\Psi}^Y$ can be derived with simple algebra. A new song can be recommended by mapping the computed audio features to the latent space. Other non-linear mapping function can be employed (e.g. by using deep neural nets), but we leave this as a future work.

Algorithm 1 Adaptive linear mapping model (ALMM)

Require: *Confidence values* tensor: $\boldsymbol{C}$, audio features: $\boldsymbol{A}$
Initialize $\boldsymbol{X}$ and $\boldsymbol{Y}$
repeat
  for $u \in N_u$ **do**
    Update $\boldsymbol{U}_u$ with equation (3)
  end for
  for $i \in N_s$ **do**
    Update $\boldsymbol{X}_i$ with equation (4)
  end for
  $\boldsymbol{\Psi}^X \leftarrow \boldsymbol{X}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T + \lambda_{\Psi^X}\boldsymbol{\Psi}^X)^{-1}$
  $\boldsymbol{X} \leftarrow \boldsymbol{\Psi}^X\boldsymbol{A}$
  for $j \in N_s$ **do**
    Update $\hat{\boldsymbol{Y}}_j$ with equation (5)
  end for
  $\boldsymbol{\Psi}^Y \leftarrow \boldsymbol{Y}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T + \lambda_{\Psi^Y}\boldsymbol{\Psi}^Y)^{-1}$
  $\boldsymbol{Y} \leftarrow \boldsymbol{\Psi}^Y\boldsymbol{A}$
until convergence
return $\boldsymbol{U}_u$, $\boldsymbol{\Psi}^X$ and $\boldsymbol{\Psi}^Y$

3. EVALUATION

3.1 Dataset

We evaluate our method on a real-world dataset collected from a regional leading music streaming service provider (anonymized for review) from October 2012 to September 2013. The dataset includes 28k users, 124k songs and 0.1 billion listening records. Each of the listening record contains a listening timestamp, song title, artist name, album name, release date and genre labels. The dataset also allows us to access to the corresponding audio files from which various audio features can be extracted. In this work, we consider as audio features the classic mel-frequency cepstral coefficients (MFCC) [5] and the more advanced audio word (AW)-based features [4] constructed by using a sparsity-enforced dictionary learning method. Finally, as shown in Table 1, we split the dataset into training set, validation set and test sets. Specifically, both the warm start (WS) and cold start (CS) settings are considered in forming the test sets. For the WS setting all users and songs can be observed in the training set, whereas for the CS setting the randomly selected 1,000 songs cannot be found in the training set at all.

3.2 Evaluation Metrics

The mean average precision (MAP) and recall rate are used for evaluation. Following Chou *et al.* [1], we also evaluate the quality of recommendation result in terms of novelty, diversity, freshness and popularity as calculated from the top-N recommended songs. Unlike MAP and recall, these metrics do not require the songs under evaluation have been listened to by the user.

- **Novelty:** measures the percentage of artists that a user is already knew, based on their listening data.
- **Diversity:** measures the entropy of the genre distribution of the recommended songs. The genre labels are obtained from the service provider.
- **Freshness:** measures the average of the release date of the recommended songs, using again the information obtained from the service provider.
- **Popularity:** measures the number of users who have listened to the song as observed in the listening data.

Notably, these measurements evaluate different aspects of a recommender system, and a combination of them may provider an estimate of a user's subjective satisfaction of the recommendation result.

3.3 Result

Table 2 shows the MAP and recall for content-based next-song recommendation. Two baseline CB methods and ALMM are implemented using the PF model with 10 latent dimension. The same linear mapping function is used in both ALMM and Oord. As Table 2 shows, ALMM outperforms the other two baseline CB methods with different audio fea-

Table 2: Accuracy of three content-based approaches, ALMM is proposed in this work. There models with two audio features is compared in term of warm start (WS) and cold start (CS)

Feature	Model	MAP@10		MAP@20		Recall@10		Recall@20	
		WS	CS	WS	CS	WS	CS	WS	CS
MFCC [5]	Forbes [2]	0.0224	0.0168	0.0287	0.0238	0.0763	0.0661	0.1702	0.1711
	Oord [6]	0.1643	0.0357	0.1727	0.0402	0.3723	0.1452	0.4961	0.2096
	ALMM	**0.1676**	**0.0409**	**0.1757**	**0.0466**	**0.3884**	**0.1530**	**0.5133**	**0.2368**
AW [4]	Forbes [2]	0.0242	0.0223	0.0311	0.0288	0.0941	0.0782	0.1961	0.1771
	Oord [6]	0.2125	0.0609	0.2192	0.0689	0.3479	0.1582	0.4523	0.2679
	ALMM	**0.2266**	**0.0680**	**0.2317**	**0.0752**	**0.3854**	**0.1670**	**0.4646**	**0.2752**

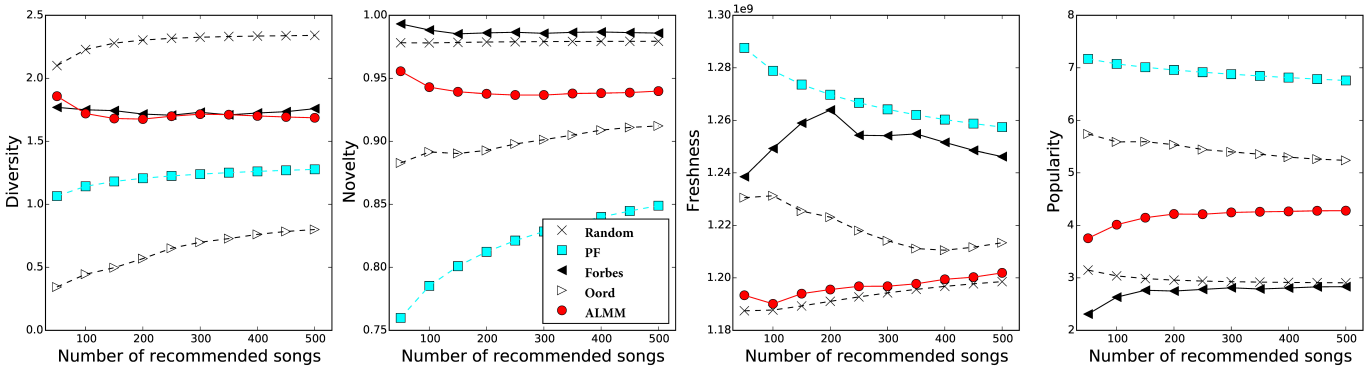

Figure 2: The performance of chose models for top-N recommendation, measured in four evaluation criteria of simulate user's satisfaction. Comparison PF and various CB approaches using AW. The recommended songs is increased by 50 from 50 to 500.

tures in both WS and CS. In contrast, Forbes yields the worst MAP and recall among the competing methods, which indicates this CB method can not work well for next-song recommendation. Moreover, this method requires more computation time than the other method. On the other hand, we can find that the Oord method has similar performance with ALMM. This is because our method is extended from Oord. However, the proposed method leads to slightly better performance measurement in both MAP and recall.

Possibly equally importantly, Figure 2 shows that the recommendation result of the proposed ALMM method exhibits higher novelty, higher novelty, lower freshness and lower popularity than the result of PF. This result suggests that ALMM can better discover new songs across different levels of popularity. In contrast, although Oord has comparable accuracy with ALMM, its result exhibits higher popularity and the lowest diversity, which may be attributed to the mismatch between the user latent vectors and the mapped item latent vectors. In sum, ALMM can obtain better accuracy, and can provide higher diversity and novelty with lower freshness and popularity, comparing to other methods.

4. CONCLUSIONS

In this paper, we address the cold start problem for next-song recommendation. The proposed algorithm captures the content-based transition preference by mining both sequential behavior and content feature simultaneously. Experimental result shows that the proposed method outperforms other CB methods in both warm start and cold start settings. Additionally, the use of content feature leads to recommendations that feature greater diversity and novelty. Although the method is evaluated only on music data, we believe it holds the promise of being applied to other domains as well.

5. REFERENCES

[1] S.-Y. Chou et al. Evaluating music recommendation in a real-world setting: On data splitting and evaluation metrics. In *Proc. ICME*, pages 1–6, 2015.

[2] P. Forbes and M. Zhu. Content-boosted matrix factorization for recommender systems: Experiments with recipe recommendation. In *Proc. ACM Recsys*, pages 261–264, 2011.

[3] Y. Hu et al. Collaborative filtering for implicit feedback datasets. In *Proc. ICDM*, pages 263–272, 2008.

[4] B. Mcfee et al. Learning content similarity for music recommendation. *IEEE Trans. Audio, Speech, and Language Processing*, 2012.

[5] M. Müller et al. Signal processing for music analysis. *J. Sel. Topics Signal Processing*, 5(6):1088–1110, 2011.

[6] A. Oord et al. Deep content-based music recommendation. In *Proc. NIPS*, pages 2643–2651. 2013.

[7] S. Rendle. Factorization machines with libFM. *ACM TIST*, 3(3):57:1–57:22, May 2012.

[8] S. Rendle et al. Factorizing personalized markov chains for next-basket recommendation. In *Proc. ACM WWW*, pages 811–820, 2010.

[9] X. Wang and Y. Wang. Improving content-based and hybrid music recommendation using deep learning. In *Proc. ACM Multimedia*, pages 627–636, 2014.

[10] X. Wu et al. Personalized next-song recommendation in online karaokes. In *Proc. ACM Recsys*, pages 137–140, 2013.