Improving Content-based and Hybrid Music Recommendation using Deep Learning

Xinxi Wang, Ye Wang
School of Computing, National University of Singapore
{wangxinxi, wangye}@comp.nus.edu.sg

ABSTRACT

Existing content-based music recommendation systems typically employ a *two-stage* approach. They first extract traditional audio content features such as Mel-frequency cepstral coefficients and then predict user preferences. However, these traditional features, originally not created for music recommendation, cannot capture all relevant information in the audio and thus put a cap on recommendation performance. Using a novel model based on deep belief network and probabilistic graphical model, we unify the two stages into an automated process that simultaneously learns features from audio content and makes personalized recommendations. Compared with existing deep learning based models, our model outperforms them in both the warm-start and cold-start stages without relying on collaborative filtering (CF). We then present an efficient hybrid method to seamlessly integrate the automatically learnt features and CF. Our hybrid method not only significantly improves the performance of CF but also outperforms the traditional feature based hybrid method.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval models; H.5.5 [**Sound and Music Computing**]: [Modeling, Signal analysis, synthesis and processing]

General Terms

Model, Algorithm, Experimentation

Keywords

music recommendation, deep learning, feature learning, probabilistic graphical model

1. INTRODUCTION

A music recommendation system automatically recommends songs that match a user's music preference from a large database. The quality of a match is influenced by many factors concerning the user (e.g., personality, emotional states, activities, social environment) and the song (e.g., music audio content, novelty, diversity).

Among song-related factors, *music audio content* is of great importance. In most cases, we like/dislike a song as a result of characteristics from its audio content, such as vocal, melody, rhythm, timbre, genre, instrument, or lyrics. Without listening to the content, we know almost nothing about the song's quality, let alone whether we would like it. Because music content largely determines our preferences, it should be able to provide good predictive power for recommendation.

However, existing music recommenders relying on music audio content usually produce unsatisfactory recommendation performance. They all follow a *two-stage* approach: extracting traditional audio content features such as Mel-frequency cepstral coefficients (MFCC), then using these features to predict user preferences [1, 2, 3]. Traditional audio content features, however, were not created for music recommendation or music related tasks (For example, MFCC was originally used for speech recognition [4]). They only became attached to music recommendation after the discovery that they can also describe high-level music concepts like genre, timbre, and melody. Using such features can result in poor recommendation performance in two ways. First, the high-level concepts cannot be described accurately due to the so-called *semantic gap* [5]. Second, even if the feature descriptions are accurate, the high-level concepts may not be essential to the user's music preferences. Therefore, traditional features could fail to take into account information relevant to music recommendation.

We believe that the key to an effective content-based music recommendation method is a set of good content features. *Manually* crafting such features is possible but time consuming and painstaking. A better approach is to combine the existing two-stage approach into a unified and automated process: features are *learnt* automatically and directly from audio content to maximize recommendation performance. Recent development in *deep learning* techniques [6] has made such a unified approach possible. In fact, people have already started using deep learning to learn features for other music tasks such as music genre classification [7] and music emotion prediction [8] with promising results.

Content methods also frequently combines *collaborative filtering* (CF), which recommends songs based on the interests of like-minded users. Most existing recommenders are based on CF because of its superior accuracy [9]. How-

MM'14, November 3–7, 2014, Orlando, Florida, USA.
Copyright 2014 ACM 978-1-4503-3063-3/14/11 ...$15.00.
http://dx.doi.org/10.1145/2647868.2654940 .

ever, as it depends solely on usage data, CF is powerless when confronted with the new-song problem — it cannot recommend songs without prior usage history. Content-based methods do not suffer from this problem because they can predict based on a song's audio content, which is usually available for online merchants. Therefore, content-based methods can rescue CF in the new-song scenario. Because CF and content-based methods take advantage of different dimensions of information, it is possible to combine them into a *hybrid method* for better predictions.

Thus motivated, we first develop a content-based model that automatically and simultaneously extracts features from audio content and makes personalized recommendations. We then develop a hybrid method to combine both CF and content features. Specifically, this paper seeks to make the following contributions:

- *Content-based method*: We develop a novel content-based recommendation model based on probabilistic graphical model and the *deep belief network* (DBN) proposed by the deep learning community [10]. It unifies feature learning and recommendation. While it does not rely on collaborative filtering, it outperforms baseline content-based models, which rely on CF, in both the *cold-start* stage and *warm-start* stage.

- *Hybrid method:* To combine CF and music content, we apply the automatically learnt audio features to an efficient hybrid model. Experimental results show that the learnt features complement CF and also outperform traditional features in hybrid methods.

The remainder of this paper is organized as follows. Section 2 briefly reviews the prevailing method in music recommendation and briefly introduces deep belief network as well as its applications in music tasks. Section 3 describes our content-based and hybrid recommendation model and discusses the baseline content-based model used in our experiments. Section 4 describes our extensive experimental evaluations. Section 5 concludes this work and discusses future research directions.

2. RELATED WORK

We will enumerate the current music recommendation techniques before introducing deep belief network and other deep learning techniques. We will then review applications of deep learning techniques in music tasks including music recommendation.

2.1 Music recommendation

Currently music recommender systems can be classified into four categories: collaborative filtering (CF), content-based methods, context-based methods and hybrid methods.

Collaborative filtering recommends songs by considering the preferences of other like-minded users. For instance, if user A and B have similar music preferences, then songs liked by A but not yet considered by B will be recommended to B. The state-of-the-art methods for performing CF are based on matrix factorization (MF), which is well summarized by [11]. In Section 3.1, we will elaborate on one of the MF method, probabilistic matrix factorization.

Content-based methods recommend songs that have similar audio content to the user's preferred songs. Most existing content-based methods first extract traditional audio features such as MFCC and then recommend based on the similarities between the feature vectors of songs. However, the similarity metrics used are often ad hoc, because they are not optimized with respect to the recommendation objective and are usually chosen from a very restrictive set of distance functions such as Euclidean distance [12, 13], Earth Mover's distance [14], or Pearson correlation distance [15, 16]. While two recent works tried to employ machine learning techniques to automatically learn a similarity metric [17, 18], they still relied on traditional features. Attempts have been made to perform feature selection or transformation on traditional features [13, 15], but they remain suboptimal as the traditional features may fail to take into account essential information.

Context-based methods recommend songs to match various aspects of the user context (e.g., activities, environment, or physiological states [3, 19]). They have become increasingly popular in recent years with the advent of sensor-rich and computationally powerful smartphones.

Hybrid methods combine two or more of the above methods. Hybrid CF and content-based methods[1] have been explored extensively in recommenders for other products such as movies [20, 21, 22, 23]. Although such approaches can potentially generalize to music recommendation, we do not use them due to efficiency issues: (1) they use full Bayesian inference [20, 22, 23] or Monte Carlo simulation [24] and are thus much slower than our algorithm; (2) they have been applied to a dataset with thousands of users and items and 1 million ratings, while our dataset has hundreds of thousands of users and items and 28 million ratings. Directly applying these algorithms on our dataset is thus not trivial.

To our knowledge, Yoshii *et al.* [2] are the first to combine CF and content-based methods in music recommendation. In this work, MFCC features were quantized into codewords and used together with rating data in the three-way aspect model, a probabilistic model, originally proposed in [25] for bibliographic recommendation. Almost concurrently, Li *et al.* [26] built a probabilistic hybrid approach to unify CF and traditional features.

While Yoshii and Li's works were promising starting points for *model-based* hybrid methods, subsequent studies all focused on content similarity based methods. Castillo [27] proposed a hybrid recommender by linearly combining the results of a content similarity based recommender and a collaborative filtering based one. Tiemann *et al.* [28] and Shruthi *et al.* [29] developed approaches that successfully fused CF and content similarities but revealed little information about the fusion process. Bu *et al.* [30] and Shao *et al.* [31] used hyper-graphs to combine usage data and content similarity information. Domingues *et al.* [32] first obtained song similarities based on CF and content features separately before integrating the two kinds of similarities into a hybrid similarity metric. Similarly, Bogdanov *et al.* [16] also combined content similarity and Last.fm's similarity, which is likely based on CF. Combining the similarities of different modalities is relatively easy and may work to some extent in practice, but the similarity metrics are usually selected in an ad hoc way.

[1] In subsequent part of this paper, we will use "hybrid method" to refer to "hybrid collaborative filtering and content-based method".

To summarize, most existing content-based methods and all hybrid methods use traditional features. This could change with the advent of deep learning, which we will introduce in the following section.

2.2 Deep learning

Deep learning methods mimic the architecture of mammalian brains. They can automatically learn features at multiple levels directly from low-level data without resorting to manually crafted features. We give a very brief introduction to *deep belief networks* (DBN), which will be used in this work, and refer the readers to Bengio *et al.* [33] for a more comprehensive review of deep learning techniques.

A deep belief network is a generative probabilistic graphical model with many layers of hidden nodes at the top and one layer of observations at the bottom. Connections are allowed between two adjacent layers but not between the same layer. Connections of the top two layers are undirected while the rest are directed. Jointly training all layers is computationally intractable, so Hinton *et al.* [10] developed an efficient algorithm to train the model layer by layer from bottom to top in a greedy manner. This unsupervised training process is usually called *pre-training.* Afterward, the DBN can be converted to a *multi-layer perceptron* (MLP) for supervised learning. This stage is called *finetune* and is usually implemented as *back-propagation.* It is also possible to directly train a MLP using back-propagation without the pre-training step, but this is prone to overfitting, especially when the MLP is deep (has more than two hidden layers). Pre-training may help by implicitly effecting a form of regularization [34].

2.3 Deep learning in music related tasks

The field of music information retrieval (MIR) has only recently begun to embrace the power of deep learning. Lee *et al.* [35] used a convolutional deep belief network to extract features in an unsupervised fashion for tasks such as music genre classification. Results show that the automatically learnt features significantly outperforms MFCC. In Hamel *et al.* [7], deep belief network was used for music genre classification and autotagging, with performance surpassing that based on MFCC and MIM feature sets. In [36, 37], Humphrey *et al.* proposed that the traditional two-stage machine learning process — feature extraction and classification/regression — should be conducted simultaneously. To classify the rhythm style of a piece of music, Pikrakis applied DBN to engineered features representing rhythmic signatures [38]. Schmidt *et al.* [39] found that DBN easily outperforms traditional features in understanding rhythm and melody based on music audio content. Other feature learning techniques like sparse coding also started to be used in music tasks. In [40], a sparse coding based approach was used to learn interpretable audio features in an unsupervised way, and good performance was achieved in music genre classification.

To the best of our knowledge, the first deep learning based approach for music recommendation was almost concurrently proposed by Oord *et al.* [41] within the last year. They first conducted matrix factorization to obtain latent features for all songs, and then used deep learning to map audio content to those latent features. Comparisons between their methods and ours will be detailed later.

Symbol	Description
u	User u
v	Song v
r_{uv}	The rating that user u gives to song v
γ_u	The latent features for u estimated by MF
y_v	The latent features for v estimated by MF
β_u	User u's preference of content features
μ	All users' common preference
x_v	The learnt content features for song v
Ω	The parameters of DBN
$\mathcal{U}$, $\mathcal{V}$	User and song sets, respectively
U, V	The number of users and songs, respectively
I	All user, song pairs in the training dataset

Table 1: Frequently used symbols

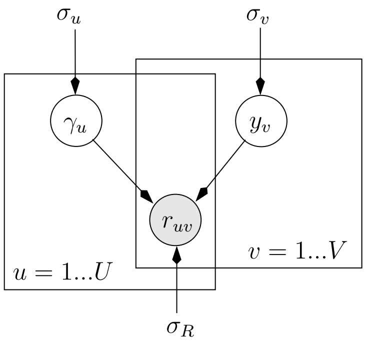

Figure 1: Probabilistic matrix factorization

To summarize, all content-based music recommendation methods except [41] and all hybrid methods are based on traditional features, which are not created for music recommendation. To enable simultaneous feature extraction and recommendation, we will build a unified model for pure content-based recommendation. We will also show that the automatically learnt features can be applied to our efficient hybrid method.

3. RECOMMENDATION MODELS

In this section, we will introduce our content-based model and hybrid model, as well as the two baseline content-based models with which to compare our models.

3.1 Collaborative filtering via probabilistic matrix factorization

Collaborative filtering is a popular recommendation method. The state-of-the-art CF methods are based on matrix factorization (MF). A MF method named probabilistic matrix factorization (PMF) [42] is used in this paper for its simplicity, accuracy, and efficiency. In addition, PMF's principled probabilistic interpretation enables it to be extended to incorporate content information more easily.

PMF assumes that each user $u \in \mathcal{U}$ and song $v \in \mathcal{V}$ can be represented as latent feature vectors γ_u and y_v, respectively. The rating that user u gives to song v is the inner product of γ_u and y_v. The training data is usually very sparse, and without regularization the model is crippled by severe overfitting. Therefore, Gaussian priors are used for both γ_u and y_v as regularization. Formally, the model is specified as the

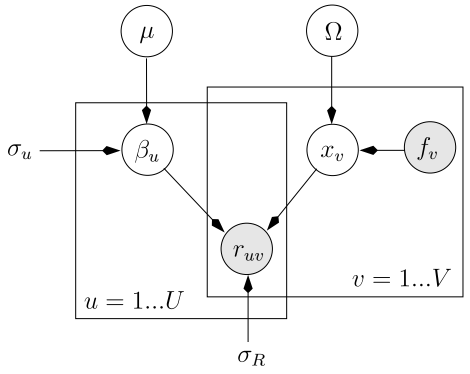

Figure 2: Hierarchical linear model with deep belief network

following[2] (see graphical representation in Figure 1):

$$r_{uv}|\gamma_u, y_v, \sigma_R \sim \mathcal{N}\left(\gamma_u' y_v, \sigma_R^2\right)$$
$$\gamma_u|\sigma_u \sim \mathcal{N}\left(\mathbf{0}, \sigma_u^2\mathbf{I}\right)$$
$$y_v|\sigma_v \sim \mathcal{N}\left(\mathbf{0}, \sigma_v^2\mathbf{I}\right)$$

The negative log-likelihood of the model can be simplified as Equation (1), where I is the user-song pairs in the training set. λ_u and λ_v are usually tuned using a validation data set [42].

$$L_{\mathrm{MF}} = \sum_{u,v\in I} \left(r_{uv} - \gamma_u' y_v\right)^2 + \lambda_u \sum_u \|\gamma_u\|^2 + \lambda_v \sum_v \|y_v\|^2 \tag{1}$$

Since a new user/song without rating data has no vector representation in the model, their ratings cannot be predicted. This cold-start problem is endemic to all CF methods. In the following sections, we will introduce our solution to the new-song problem.

3.2 Content-based Music Recommendation

3.2.1 *Hierarchical linear model with deep belief network (HLDBN)*

We assume that the audio content of song v is f_v, and its automatically learnt feature vector is x_v. User u's music preference is represented as a vector β_u. The rating that u gives to song v, denoted as r_{uv}, is the inner product of x_v and β_u. We use μ to represent all users' common music preference, which is the mean of all users' β_u-s. The model (Figure 2) is formulated as:

$$r_{uv} \sim \mathcal{N}\left(\beta_u' x_v, \sigma_R^2\right)$$
$$\beta_u \sim \mathcal{N}\left(\mu, \sigma_u^2\mathbf{I}\right)$$
$$x_v = \mathrm{DBN}\left(f_v; \Omega\right)$$

σ_u indicates the variance of user preferences. The smaller the σ_u, the more similar the user preferences are to the common preference μ and the more strongly β_u is regularized. The Gaussian prior for β_u models user common interests as *one* cluster. However, users of different genders, ages and different culture backgrounds could form different groups. To capture this grouping effect, we could change the single Gaussian prior to a mixture of Gaussians. We tried such a prior and used Monte Carlo Expectation Maximization to estimate the parameters, but it resulted in overfitting. Therefore, we chose single Gaussian as the prior.

Automatic learning of features x_v from music content f_v is achieved by deep belief network (DBN), which is briefly introduced in Section 2.2 and 2.3. DBN can be treated as a very flexible deterministic function that maps f_v to x_v. It has hundreds of thousands, perhaps even millions, of parameters (denoted as Ω) to be learnt from training data. We assume that r_{uv} follows a normal distribution to account for the noise in user ratings.[3]

Learning - Maximum Likelihood Estimation (MLE) is used to train the model. The negative log-likelihood of the model is shown in Equation (2), where irrelevant constants are omitted. The hyperparameter λ is the ratio σ_u^2/σ_R^2, with a larger λ indicating stronger regularization.

$$L_{\mathrm{HLDBN}} = \sum_{u,v\in I} \left(r_{uv} - \beta_u' \mathrm{DBN}\left(f_v, \Omega\right)\right)^2 + \lambda \sum_u \|\beta_u - \mu\|^2 \tag{2}$$

Since Ω consists of a large amount of parameters, directly optimizing L_{HLDBN} using gradient descent could easily overfit. Following the DBN training procedure established in [10], we first pre-train the DBN as stacked layers of Restricted Boltzmann Machines in an unsupervised fashion and then optimize L_{HLDBN} using mini-batch stochastic gradient descent, where the gradient descent part of DBN is implemented as back-propagation.[4]

Unlike traditional two-stage methods, our model automatically and simultaneously optimizes audio features (x_v) and user preference parameters (β_u-s). This provides a unified and more principled method to content-based recommendation.

Prediction - After the learning phase, the rating that user u gives to song v can be estimated as $\hat{r}_{uv} = \gamma_u' \mathrm{DBN}(f_v, \Omega)$. As the predictions are based on audio content, new songs can be recommended accurately as well.

3.2.2 *Baseline models*

We now turn our attention to the two content-based approaches proposed in Oord *et al.* and hereby used as our baseline methods [41]. The models are based on convolutional neural network (CNN), another popular deep learning method. To make their approach directly comparable with ours, we replace CNN with DBN while keeping the other parts unchanged.

Content-based baseline model 1 (CB1) - This model first uses PMF to learn latent features γ_u and y_v for all users and songs and then trains a DBN to map from audio content to the latent features y_v. Formally, the objective can be formulated as:

$$\min_{\Omega} \sum_v \left(y_v - \mathrm{DBN}\left(f_v, \Omega\right)\right)^2 \tag{3}$$

[2] $\mathcal{N}(a, b)$ is the normal distribution with mean a and variance b. $x \sim p$ indicates that x satisfies the distribution p or x is drawn/generated from p.

[3] The normal distribution may be replaced with a softmax or probit model. In this paper, we follow PMF and use the normal distribution to keep the model clean.

[4] For the following DBN-based models, the same training approach is used.

Let $x_v = \text{DBN}(f_v, \Omega)$; the rating that user u gives to song v can be predicted as $\hat{r}_{uv} = \gamma_u' x_v$. This model, however, fails because of a fundamental flaw shown in Theorem 1.

THEOREM 1. *Model CB1 does not minimize the sum of squared errors of predicted ratings.*

PROOF. Let $\epsilon_v = y_v - x_v$. The optimization objective Equation (3) is equivalent to

$$\min_{\Omega} \sum_v \|\epsilon_v\|^2 \tag{4}$$

Instead of predicting the latent features, our *true objective* is to predict ratings, so we actually need to minimize the sum of squared errors of predicted ratings:

$$\min_{\Omega} \sum_{u,v \in I} \left(r_{uv} - \gamma_u' x_v\right)^2 \tag{5}$$

which can be transformed as:

$$\begin{aligned} &\min_{\Omega} \sum_{u,v \in I} \left(r_{uv} - \gamma_u' x_v\right)^2 \\ =&\min_{\Omega} \sum_{u,v \in I} \left(r_{uv} - \gamma_u' \left(y_v - \epsilon_v\right)\right)^2 \\ =&\min_{\Omega} \sum_v \epsilon_v \left(\sum_{u:u,v \in I} \gamma_u \gamma_u'\right) \epsilon_v' + 2 \sum_{u:u,v \in I} \gamma_u' \left(r_{uv} - \gamma_u' y_v\right) \epsilon_v \end{aligned} \tag{6}$$

Since ϵ_v is not constrained because MLPs are universal approximators [43], we can see that Equation (6) and (4) have different optimal solutions. □

The original model in Oord et al. [41] uses weighted sum of squared errors. Following the same approach, we can prove that CB1 does not minimize the weighted version, either.

Content-based baseline model 2 (CB2) - This is the other model proposed by Oord et al. [41]. It is presented as the following,

$$\min_{\Omega} \sum_{u,v \in I} \left(r_{uv} - \gamma_u' \text{DBN}\left(f_v, \Omega\right)\right)^2 \tag{7}$$

where γ_u is obtained from MF beforehand. Rating r_{uv} is predicted as $\gamma_u' x_v$, where $x_v = \text{DBN}(f_v, \Omega)$.

This model uses the correct objective and thus does not have the issue of CB1 discussed in Theorem 1. However, it lacks regularization on the parameters, which may cause overfitting. We will show this empirically in Section 4.5.

Another issue of both CB1 and CB2 is that they are directly based on the results of MF and thus their prediction results are strongly correlated with the collaborative filtering (CF) results. As we will show in Section 4.5 and 4.6, this hinders us from combining CB1 or CB2 with CF to form an effective hybrid approach.

3.3 Hybrid CF and content-based music recommendation

Collaborative filtering and content-based methods use different information. To fuse all the information available for more accurate predictions, we can combine the two in a hybrid method.

Information fusion has been studied extensively in other domains such as sensor fusion and multimedia information fusion. There are mainly two approaches for our problem. Decision fusion combines the prediction results from existing CF and content-based methods. On the other hand, data fusion develops a new unified model to incorporate both CF and audio content. Our hybrid method is based on the latter, but it also uses the features learnt by HLDBN.

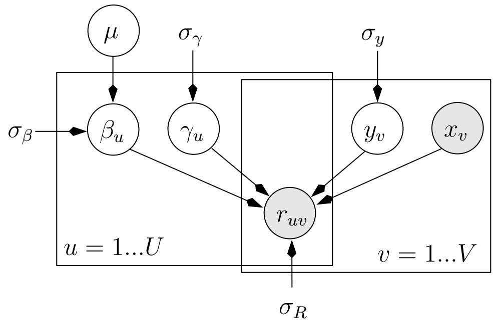

Figure 3: Hybrid recommendation

In our hybrid model (Figure 3), we assume that the audio features x_v for every song is already known. γ_u, y_v and β_u are not directly adopted from the results of PMF and HLDBN but need to be jointly learnt from data. Rating r_{uv} is predicted by the sum of the CF part $\gamma_u' y_v$ and the content part $\beta_u' x_v$. The priors for γ_u and y_v are set following the PMF model, and β_u the HLDBN model.

$$\begin{aligned} r_{uv} | \beta_u, x_v, \gamma_u, y_v, \sigma_R &\sim \mathcal{N}\left(\beta_u' x_v + \gamma_u' y_v, \sigma_R^2\right) \\ \beta_u | \sigma_\beta &\sim \mathcal{N}\left(\mu, \sigma_\beta^2 \mathbf{I}\right) \\ \gamma_u | \sigma_u &\sim \mathcal{N}\left(\mathbf{0}, \sigma_\gamma^2\right) \\ y_v | \sigma_v &\sim \mathcal{N}\left(\mathbf{0}, \sigma_y^2\right) \end{aligned} \tag{8}$$

The negative log-likelihood can be simplified as the following, where $\lambda_\beta = \sigma_R^2/\sigma_\beta^2$, $\lambda_\gamma = \sigma_R^2/\sigma_\gamma^2$, and $\lambda_y = \sigma_R^2/\sigma_y^2$.

$$\begin{aligned} L_{\text{Hybrid}} = &\sum_{u,v \in I} \left(r_{uv} - \beta_u' x_v - \gamma_u' y_v\right)^2 + \lambda_\beta \|\beta - \mu\|_F^2 \\ &+ \lambda_\gamma \|\gamma\|_F^2 + \lambda_y \|y\|_F^2 \end{aligned}$$

L_{Hybrid} is not a convex function, but if we fix any three of β_u, μ, γ_u, and y_v, it is convex and the optimal solution can be obtained in closed form. We thus optimize L_{Hybrid} using the alternative least square (ALS) algorithm: we first set the derivatives of L_{Hybrid} with respect to each of the four parameters to zero and solve the equations, which results in the following four updating formulas. We then iterate them until L_{Hybrid} converges or until the prediction performance on a validation set reaches the highest point.

$$\begin{aligned} \beta_u &\leftarrow \left(\sum_{v:u,v \in I} x_v x_v' + \lambda_\beta \mathbf{I}\right)^{-1} \left(\lambda_\beta \mu + \sum_{v:u,v \in I} \left(r_{uv} - \gamma_u' y_v\right) x_v\right) \\ \gamma_u &\leftarrow \left(\sum_{v:u,v \in I} y_v y_v' + \lambda_\gamma \mathbf{I}\right)^{-1} \left(\sum_{v:u,v \in I} \left(r_{uv} - \beta_u' x_v\right) y_v\right) \\ y_v &\leftarrow \left(\sum_{u:u,v \in I} \gamma_u \gamma_u' + \lambda_y \mathbf{I}\right)^{-1} \left(\sum_{u:u,v \in I} \left(r_{uv} - \beta_u' x_v\right) \gamma_u\right) \\ \mu &\leftarrow \frac{1}{U} \sum_{u \in \mathcal{U}} \beta_u \end{aligned}$$

To achieve faster convergence, γ_u, y_v are first initialized using PMF.

We could create a pure data fusion model by adding $x_v = \text{DBN}(f_v, \Omega)$ after Equation 8, and optimizing Ω jointly with other parameters. However, we found its performance inferior to the one above.

4. EXPERIMENTS

4.1 Dataset

Deep belief network has a large number of parameters, and a large amount of data is required to adequately train such a model. We chose The Echo Nest Taste Profile Subset [9] because it is the largest publicly available music recommendation dataset as far as we know. The original dataset has $1,019,318$ users, $384,546$ songs, and $48,373,586$ listening histories. We were able to crawl preview audio clips with length of about half a minute from 7digital[5] for $282,508$ of the songs. We selected the top $100,000$ users mainly to reduce the training time.

Implicit feedback - From the Taste Profile Subset, we know the songs that a user has listened to, so the dataset can be presented as a set of (user, song) pairs. We assign a rating of 1 to each pair and use them as the positive samples. To generate negative samples, we use the well-established User-Oriented Sampling method built in [44]: for user u who listened to songs $\mathcal{V}_u$, we randomly sampled $|\mathcal{V}_u|$ songs from $\mathcal{V}\backslash\mathcal{V}_u$ and assign a rating 0 to each generated (user, song) pair. We now have equal number of positive and negative samples for every user.

Instead of using the sampling method described above, HLDBN could theoretically use the weighted matrix factorization method proposed in Hu *et al.* [45] to directly handle the implicit feedback. While the method may be more accurate, the computational overhead is prohibitive: there will be about 2.83×10^{10} rating data points, which can make all algorithms about 1000 times slower and take years to finish.

Table 2 gives the statistics of the final dataset. The density of the rating matrix is only 0.1%.

Splitting the dataset - The dataset was then split into 5 disjoint sets: the training set, warm-start validation/test sets, and cold-start validation/test sets. All users and songs in the warm-start sets need to be in the training set. To simulate the new-song problem, songs in the cold-start validation/test sets cannot exist in the training set, while all users in the cold-start sets still need to be in the training set because the new user problem is not our focus. The statistics of the five datasets are shown in Table 2, where WS and CS stand for warm-start and cold-start, respectively.

Audio content preprocessing - We first converted all audio clips to WAV files with mono channel, 8kHz sampling rate, and 16 bit depth. We then randomly sampled a 5-second continuous segment from each audio clip, because directly using the half-minute clips requires too much memory and computation while segments shorter than 5 seconds may lose too much information. We next converted each 5-second segment into a 166×120 spectrogram (30ms window, no overlap). PCA was then used to transform the spectrograms into vectors whose dimensions were ranked according to their significance. The top-K dimensions were finally normalized to have zero mean and unit variance and fed into DBN. The normalization step is required because we use Gaussian-Bernoulli RBM for the DBN's input layer [46]. K, the dimensionality of f_v and the number of nodes of the DBN's input layer, is determined by a validation step.

	# of users	# of songs	# of ratings
Total	$100,000$	$282,508$	$28,258,926$
Train	$100,000$	$262,508$	$18,382,954$
WS Valid	$100,000$	$262,454$	$3,939,204$
WS Test	$100,000$	$262,457$	$3,939,206$
CS Valid	$99,963$	$10,000$	$1,025,654$
CS Test	$99,933$	$10,000$	$971,908$

Table 2: Dataset statistics

4.2 Implementation and Training of deep belief network

We implemented our DBN using Theano[6], because it supports convenient GPU programming and automatic symbolic differentiation. Since our input for DBN is continuous, we used the Gaussian-Bernoulli RBM for the input layer and binary RBMs for the rest [46].

Training DBN on CPUs is extremely slow. GPUs with large memory are thus indispensable in the deep learning experiments. Training and testing of every model takes three to four days using a single GPU with 6GB GPU memory. Since DBN has many hyperparameters that could have great impact on the prediction performance, tuning seems unavoidable at this stage. Sequentially trying each configuration of the hyperparameters on a single GPU is too time consuming. We thus utilized a GPU cluster with 15 computing nodes, each of which containing two Tesla M2090 GPU cards.

Mini-batch stochastic gradient descent was used as the training algorithm. We cannot transfer all data into one GPU because of its memory limit. Sequentially transferring one batch after computing the previous batch is slow because of the low bandwidth of the bus between the GPU memory and main memory. Our solution is to use multithreading to enable computing and transferring next batch at the same time.

4.3 Evaluation Metrics

The Root Mean Square Error (RMSE) metric was used to evaluate most models in this paper. It is defined as:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(\hat{r}_i - r_i)^2}{N}}$$

where r_i, $\hat{r}_i$ are the true and predicted ratings, respectively. We prefer RMSE to the truncated mAP used in the million song dataset challenge [9] because our models are regression models, for which RMSE is a more accurate and sensible metric [9]. Moreover, RMSE is feasible in our case because the sampling step in the preprocessing of the dataset described in Section 4.1 have converted all implicit feedback into explicit ratings.

For models which rank songs instead of predicting ratings, we still resort to truncated mAP. mAP was originally widely used in information retrieval to measure the ranking quality of search results. Suppose the system recommends user u a

[5] http://7digital.com

[6] http://deeplearning.net/software/theano/

list of songs $l_{u,1}, l_{u,2} \ldots l_{u,M}$, we first define the precision-at-k (P_k) metric as:

$$P_k(u, l_u) = \frac{1}{k} \sum_{i=1}^{k} r_{u,l_{u,i}}$$

Then we define the average precision as the following,

$$\mathrm{AP}(u, l_u) = \frac{1}{n_u} \sum_{k=1}^{M} r_{u,l_{u,k}} P_k(u, l_u)$$

where n_u is the number of songs preferred by user u, i.e. $n_u = \sum_{v=1}^{V} r_{u,v}$. Finally, mAP is defined as:

$$\mathrm{mAP} = \frac{1}{U} \sum_{u=1}^{U} \mathrm{AP}(u, l_u)$$

4.4 Probabilistic Matrix Factorization

Because CB1, CB2 and the hybrid method all depend on the results of PMF, we trained several PMF models with different configurations using the Alternative Least Square algorithm. The training procedure was stopped when the performance on the warm-start validation set converged. We found that the best performance on the validation set was achieved when the dimensionality of the latent features was 100 and $\lambda_u = \lambda_v = 4$. Further increasing the dimensionality of the latent features brought little improvement.

The results for CF are shown in Table 4. We should note that a rating predictor which randomly generates 0s or 1s have RMSE = 0.707, and a mean predictor which constantly gives 0.5s has RMSE = 0.5.

There is no result for PMF on the cold-start validation/test sets, as PMF cannot recommend during the cold-start stage (see Section 3.1).

4.5 Content-based music recommendation

Comparisons between deep learning based methods and traditional features (e.g. MFCC) based methods have been conducted in [41]. We avoid repeating those comparisons and only compare HLDBN with CB1 and CB2.

Because the objectives for the warm-start and cold-start scenarios are different, we discuss them separately in the following two sections.

4.5.1 Warm-start

Evaluating the performance of a content-based model in the warm-start stage is important for two reasons. First, the warm-start stage is a crucial stage to a recommender. Second, a content-based model performing well in the warm-start stage would serve as a better building block for a good hybrid model, whose performance in the warm-start stage is determined by both the collaborative filtering part and the content part. In the warm-start stage, all songs in the validation/test sets are in the training set. Therefore, whether the content-based model generalizes to new songs or not is not of our focus.

To determine the structure of DBN, we tried different number of layers as well as different number of nodes for each layer. For HLDBN, we finally used DBN with four layers (the input layer included), each containing 500 nodes. Increasing the number of nodes of each layer does not produce better results. Unsupervised pre-training was conducted for 200 iterations. The mini-batch size for both pre-training and finetune is 5000. The learning rate for the Gaussian-Bernoulli RBM is 5×10^{-5} and binary RBM 10^{-2}. The finetune learning rate of the supervised training stage is 0.5, and the regularization parameter $\lambda = 0.1$. The finetune process was stopped when the model's predictive performance on the warm-start validation set started to drop.

	Warm start		Cold start	
	Valid	Test	Valid	Test
PMF	0.270	0.270	-	-
HLDBN	**0.323**	**0.323**	**0.477**	**0.478**
CB1	0.679	0.679	0.688	0.669
CB2	0.325	0.325	0.495	0.495
Mean	0.500	0.500	0.500	0.500

Table 3: Predictive performance of CF and content-based methods using DBN (Root Mean Squared Error).

For CB1 and CB2, the number of nodes of their output layer is determined by the dimensionality of the PMF's latent features, i.e. 100 (see Section 4.4). Other layers use the same configuration as HLDBN.

The results of HLDBN, CB1, and CB2 are shown in Table 3. Although HLDBN only slightly outperforms CB2, we should notice that while CB2 is trained based on the results of PMF, HLDBN is a unified model and does not rely on PMF, which makes it easier to train and more principled.

The results also show that CB1 has RMSE larger than 0.5, which is worse than the trivial mean predictor. In fact, the RMSE of CB1 on the training set is also as large as 0.7, and increasing the size of DBN does not lead to improvement. These observations support our assertion in Section 3.2.2 that CB1 used the incorrect objective.

4.5.2 Cold-start

The major practical advantage of content-based methods over CF is that content-based methods work even in the new-song scenario. Thus an effective content-based model should generalize well to new songs.

Most experimental settings are the same as those in the warm-start evaluation except the configuration of DBN. Even with pre-training, large DBN is prone to overfitting. We tried many configurations and decided on using four layers, each of which contains 300 nodes. Increasing the number of input nodes makes the model overfit.

The results are shown in Table 3. We can see that HLDBN outperforms CB1 and CB2 significantly. CB1 has very poor results due to its incorrect objective function. CB2 performs only slightly better than the mean predictor. We tried to reduce the size of its DBN and also applied the deep convolutional neural network directly on the spectrogram following the same settings as [41] (the RMSEs of CB2+CNN on WS Valid, WS Test, CS Valid, and CS Test are 0.325, 0.326, 0.492, and 0.492, respectively), but there were no significant changes, which suggests that CB2 does not generalize well. This could be due to the lack of proper regularization. Therefore, among the three models, only HLDBN generalizes to new songs.

4.6 Hybrid music recommendation

The focus of our hybrid method is to boost the recommendation performance in the warm-start stage instead of

	WS Valid	WS Test
Hybrid w/ HLDBN	**0.255**	**0.255**
Hybrid w/ CB2	0.270	0.270

Table 4: Predictive performance of our hybrid method with the features learnt by our HLDBN model and the baseline CB2 model (Root Mean Squared Error). WS stands for warm-start.

	WS Valid	WS Test
PMF	0.0109	0.0110
Hybrid w/ HLDBN	**0.0132**	**0.0131**
AM w/ traditional features	0.0108	0.0108
AM w/ features from HLDBN	0.0123	0.0120

Table 5: Comparison between hybrid methods using features learnt from HLDBN and traditional features (mean Average Precision).

solving the new-song problem, for which we can simply fall back to HLDBN. It is also possible to build a model to handle both scenarios seamlessly, and we leave it as future work.

As CB1 performs worse than the trivial mean predictor, it makes little sense to train a hybrid model based on its learnt features. We thus only consider the hybrid methods based on the features learnt by HLDBN and CB2. Table 4 shows that the hybrid approach based on CB2's features does not bring any improvement over PMF's results shown in Table 3. This is because the content features learnt by CB2 are highly correlated with the latent features from PMF and do not provide much new information. On the other hand, HLDBN does not rely on PMF, and its learnt features have incorporated audio content information that PMF fails to take into account. The results show that HLDBN performs significantly better than PMF.

To show that the learnt features are better than traditional features in hybrid music recommendation, the aspect model (AM, introduced in Section 2.1), one of the two existing model-based hybrid music recommenders (Section 2.1), was chosen as the baseline. Because AM ranks songs instead of predicting ratings, we switched our evaluation metric from RMSE to truncated mAP, for which the top-500 recommended songs were considered.

For AM, we first extracted a rich set of traditional features. Marsyas [47] was used on the 30-second WAV files described in Section 4.1. A window size of 512 was used without overlapping. Descriptions about the features are shown in Table 6. Because AM cannot handle continuous features directly, we built a codebook using k-means based on 8 million feature vectors from $10,000$ randomly sampled songs. K-means was used instead of GMM in [2] mainly for better efficiency. Hadoop[7] was used to handle the large amount of data and computation. Finally, feature vectors of each song were quantized as codewords, which were further aggregated into a vector with each element representing occurrences of the corresponding codeword. Other parts remain the same as [2]. To use HLDBN's learnt features in AM, we also quantized the learnt features and aggregated each song's codewords into one vector.

Feature name	Description
MFCC	Mel-Frequency Cepstral Coefficients. It models the auditory perception system and is widely used in speech and music domain.
Zero Crossings	The rate of sign-changes along a signal.
Spectral Centroid	The "center of mass" of the spectrum. Measures the brightness of the sound.
Spectral Flux	The squared change in normalized amplitude between two consecutive time frames. It measures how much the sound changes between frames.
Spectral Rolloff	Measures the amount of the right-skewness of the power spectrum.
Spectral Flatness Measure	Measures how much the audio signal sounds like a tone instead of noise.
Spectral Crest Factor	Another measure of noisiness. Similar to Spectral Flatness Measure.
Chroma	Pitch based feature. It projects the spectrum into 12 bins, representing the 12 distinct pitches of the chromatic musical scale.
Tempo	Beats per minute

Table 6: Traditional audio features used

The results of our hybrid method and AM are shown in Table 5. We can see that AM with traditional features performs slightly worse than PMF. However, AM performs significantly better with the features learnt by HLDBN. This suggests that the automatically learnt features are more effective than the traditional features in hybrid music recommendation.

In addition, our hybrid method performs significantly better than AM. The possible reasons could be: (1) our model has regularization terms but AM does not; (2) our method can directly use the features, but AM has to quantize feature vectors, which results in information loss.

5. CONCLUSION

In this paper, we have described a novel model for content-based music recommendation based on deep belief network and probabilistic graphical model. Instead of splitting feature extraction and recommendation into separate steps, our model unifies them in an automated process. Compared with existing deep learning based models, our model outperforms them in both the warm-start and cold-start stages without relying on collaborative filtering. Based on the automatically learnt features, we created a hybrid collaborative filtering and content-based recommendation model, which not only significantly improves the performance of CF but also outperforms the traditional feature based hybrid method.

As deep learning strives to provide a model for human cognition, it has the potential to reveal many secrets behind our preferences for music. Our study on HLDBN serves as a mere starting point to tap into that potential. One practical future direction could be to further improve the recommendation performance by explicitly modeling the temporal structure of music content using deep recurrent neural network [48]. Another interesting direction could be to interpret the automatically learnt features to discover interesting characteristics of music. Also, it may be interesting to take advantage of recent advances in matrix factorization (e.g.

[7] http://hadoop.apache.org/

Lee *et al* [49]) by combining them with the deep learning methods.

6. ACKNOWLEDGMENT

This project is funded by the National Research Foundation (NRF) and managed through the multi-agency Interactive & Digital Media Programme Office (IDMPO) hosted by the Media Development Authority of Singapore (MDA) under Centre of Social Media Innovations for Communities (COS- MIC). We are also grateful to Haotian Fang for proofreading the manuscript.

7. REFERENCES

[1] P. Cano, M. Koppenberger, and N. Wack, "Content-based music audio recommendation," in *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, (New York, NY, USA), pp. 211–212, ACM, 2005.

[2] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno, "Hybrid Collaborative and Content-based Music Recommendation Using Probabilistic Model with Latent User Preferences," in *ISMIR*, pp. 296–301, 2006.

[3] X. Wang, D. Rosenblum, and Y. Wang, "Context-Aware Mobile Music Recommendation for Daily Activities," in *ACM Multimedia 2012*, Oct. 2012.

[4] P. Mermelstein, "Distance measures for speech Recognition–Psychological and instrumental," in *Joint Workshop on Pattern Recognition and Artificial Intelligence*, 1976.

[5] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, "Content-Based Music Information Retrieval: Current Directions and Future Challenges," *Proceedings of the IEEE*, vol. 96, pp. 668–696, Mar. 2008.

[6] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," Oct. 2012.

[7] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks," in *11th International Society for Music Information Retrieval Conference*, 2010.

[8] E. M. Schmidt and Y. E. Kim, "Learning emotion-based acoustic features with deep belief networks," in *ISMIR*, 2011.

[9] B. McFee, T. Bertin-Mahieux, D. P. W. Ellis, and G. R. G. Lanckriet, "The million song dataset challenge," in *21st International Conference Companion on World Wide Web*, pp. 909–916, 2012.

[10] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, pp. 1527–1554, July 2006.

[11] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, vol. 42, pp. 30–37, Aug. 2009.

[12] H. C. Chen and A. L. P. Chen, "A Music Recommendation System Based on Music Data Grouping and User Interests," in *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, (New York, NY, USA), pp. 231–238, ACM, 2001.

[13] B. Zhang, J. Shen, Q. Xiang, and Y. Wang, "CompositeMap: a Novel Framework for Music Similarity Measure," *SIGIR*, 2009.

[14] B. Logan and A. Salomon, "A Content-Based music similarity function," tech. rep., Cambridge Research Laboratory, 2001.

[15] D. Bogdanov, M. Haro, F. Fuhrmann, E. Gómez, and P. Herrera, "Content-based music recommendation based on user preference examples," in *The 4th ACM Conference on Recommender Systems. Workshop on Music Recommendation and Discovery (Womrad 2010)*, 2010.

[16] D. Bogdanov, M. Haro, F. Fuhrmann, A. Xambó, E. GóMez, and P. Herrera, "Semantic audio content-based music recommendation and visualization based on user preference examples," *Inf. Process. Manage.*, vol. 49, pp. 13–33, Jan. 2013.

[17] B. McFee, L. Barrington, and G. Lanckriet, "Learning content similarity for music recommendation," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, pp. 2207–2218, Oct. 2012.

[18] N.-H. Liu, "Comparison of content-based music recommendation using different distance estimation methods," *Applied Intelligence*, vol. 38, pp. 160–174, June 2013.

[19] M. Schedl and D. Schnitzer, "Location-Aware Music Artist Recommendation," in *MultiMedia Modeling* (C. Gurrin, F. Hopfgartner, W. Hurst, H. Johansen, H. Lee, and N. O'Connor, eds.), vol. 8326 of *Lecture Notes in Computer Science*, pp. 205–213, Springer International Publishing, 2014.

[20] I. Porteous, A. Asuncion, and M. Welling, "Bayesian Matrix Factorization with Side Information and Dirichlet Process Mixtures," in *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10).*

[21] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks," *International Journal of Approximate Reasoning*, vol. 51, pp. 785–799, Sept. 2010.

[22] H. Shan and A. Banerjee, "Generalized Probabilistic Matrix Factorizations for Collaborative Filtering," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pp. 1025–1030, IEEE, Dec. 2010.

[23] S. Park, Y. D. Kim, and S. Choi, "Hierarchical Bayesian Matrix Factorization with Side Information," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI'13, pp. 1593–1599, AAAI Press, 2013.

[24] D. Agarwal and B. C. Chen, "Regression-based latent factor models," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, (New York, NY, USA), pp. 19–28, ACM, 2009.

[25] R. Popescul and L. H. Ungar, "Probabilistic models for unified collaborative and content-based recommendation in sparsedata environments," in *UAI 2011*, 2001.

[26] Q. Li, S. H. Myaeng, and B. M. Kim, "A probabilistic music recommender considering user opinions and

audio features," *Information Processing and Management*, vol. 43, pp. 473–487, Mar. 2007.

[27] H. S. Del Castillo, "Hybrid Content-Based Collaborative-Filtering music recommendations," Master's thesis, University of Twente, The Netherlands, 2007.

[28] M. Tiemann and S. Pauws, "Towards ensemble learning for hybrid music recommendation," in *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, (New York, NY, USA), pp. 177–178, ACM, 2007.

[29] J. Shruthi, S. Sneha, U. R. Shetty, and D. Jayalakshmi, "A hybrid music recommender system.".

[30] J. Bu, S. Tan, C. Chen, C. Wang, H. Wu, L. Zhang, and X. He, "Music recommendation by unified hypergraph: Combining social media information and music content," in *Proceedings of the International Conference on Multimedia*, MM '10, (New York, NY, USA), pp. 391–400, ACM, 2010.

[31] B. Shao, D. Wang, T. Li, and M. Ogihara, "Music Recommendation Based on Acoustic Features and User Access Patterns," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 17, pp. 1602–1611, Nov. 2009.

[32] M. A. Domingues, F. Gouyon, A. M. Jorge, J. P. Leal, J. . Vinagre, L. Lemos, and M. Sordo, "Combining usage and content in an online music recommendation system for music in the long-tail," in *Proceedings of the 21st International Conference Companion on World Wide Web*, WWW '12 Companion, (New York, NY, USA), pp. 925–930, ACM, 2012.

[33] Y. Bengio, "Learning Deep Architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, pp. 1–127, Jan. 2009.

[34] D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, and S. Bengio, "Why Does Unsupervised Pre-training Help Deep Learning?," *J. Mach. Learn. Res.*, vol. 11, pp. 625–660, Mar. 2010.

[35] H. Lee, Y. Largman, P. Pham, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in Neural Information Processing Systems*, 2009.

[36] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Moving Beyond Feature Design: Deep Architectures and Automatic Feature Learning in Music Informatics," in *13th International Society for Music Information Retrieval Conference*, 2012.

[37] E. Humphrey, J. Bello, and Y. LeCun, "Feature learning and deep architectures: new directions for music informatics," *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 461–481, 2013.

[38] A. Pikrakis, "A deep learning approach to rhythm modelling with applications," in *6th International Workshop on Machine Learning and Music (MML13)*, 2013.

[39] E. M. Schmidt and Y. E. Kim, "Learning rhythm and melody features with deep belief networks," in *ISMIR*, 2013.

[40] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun, "Unsupervised Learning of Sparse Features for Scalable Audio Classification," in *International Society for Music Information Retrieval Conference*, 2011.

[41] A. van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *NIPS*, 2013.

[42] R. Salakhutdinov and A. Mnih, "Probabilistic Matrix Factorization," in *Advances in Neural Information Processing Systems*, 2008.

[43] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, Jan. 1989.

[44] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-Class Collaborative Filtering," in *Data Mining, 2008. ICDM. Eighth IEEE International Conference on*, vol. 0, (Los Alamitos, CA, USA), pp. 502–511, IEEE, Dec. 2008.

[45] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, vol. 0 of *ICDM '08*, (Washington, DC, USA), pp. 263–272, IEEE Computer Society, Dec. 2008.

[46] G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines," in *Neural Networks: Tricks of the Trade* (G. Montavon, G. Orr, and K.-R. Müller, eds.), vol. 7700 of *Lecture Notes in Computer Science*, pp. 599–619, Springer Berlin Heidelberg, 2012.

[47] G. Tzanetakis and P. Cook, "MARSYAS: a framework for audio analysis," *Org. Sound*, vol. 4, pp. 169–175, Dec. 1999.

[48] M. Hermans and B. Schrauwen, "Training and analyzing deep recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2013.

[49] J. Lee, S. Kim, G. Lebanon, and Y. Singer, "Local Low-Rank matrix approximation," in *Proceedings of the 30th Annual International Conference on Machine Learning*, 2013.