CONTENT-AWARE COLLABORATIVE MUSIC RECOMMENDATION USING PRE-TRAINED NEURAL NETWORKS

Dawen Liang, Minshu Zhan, and Daniel P. W. Ellis

LabROSA, Dept. of Electrical Engineering
Columbia University, New York
{dliang@ee., mz2468@, dpwe@ee.}columbia.edu

ABSTRACT

Although content is fundamental to our music listening preferences, the leading performance in music recommendation is achieved by collaborative-filtering-based methods which exploit the similarity patterns in user's listening history rather than the audio content of songs. Meanwhile, collaborative filtering has the well-known "cold-start" problem, i.e., it is unable to work with new songs that no one has listened to. Efforts on incorporating content information into collaborative filtering methods have shown success in many non-musical applications, such as scientific article recommendation. Inspired by the related work, we train a neural network on semantic tagging information as a content model and use it as a prior in a collaborative filtering model. Such a system still allows the user listening data to "speak for itself". The proposed system is evaluated on the Million Song Dataset and shows comparably better result than the collaborative filtering approaches, in addition to the favorable performance in the cold-start case.

1. INTRODUCTION

Music recommendation is an important yet difficult task in music information retrieval. A recommendation system that accurately predicts users' listening preferences bares enormous commercial value. However, the high complexity and dimensionality of music data and the scarcity of user feedback makes it difficulty to create a successful music recommendation system.

Two primary approaches exist in recommendation: collaborative filtering and content-based methods. For music, the state-of-the-art recommendation results have been achieved by collaborative filtering methods, which requires only information on users' listening history rather than the musical content for recommendation. The central assumption of this model is that a user is likely to accept a song that is liked by users who have similar taste. A major category of collaborative filtering approaches is based on latent factor model. It assumes that a low-dimensional representation exists for both users and songs such that the compatibility between a user and a song, modeled as their inner product in this latent space, predicts the user's fondness of the song. In the case that user feedback is *implicit* (e.g., whether or not the user has listened to a particular song), the weighted matrix factorization from Hu *et al.* [6] works particularly well. Details regarding collaborative filtering will be further discussed in Section 2.1.

On the other hand, modeling musical content for the purpose of taste prediction is difficult due to the structural complexity present in music data which is hard to capture by simple models. Deep learning has shown its power in various pattern recognition tasks with its capability of extracting hierarchical representations from raw data. In music recommendation, van den Oord *et al.* [13] have experimented with neural networks on predicting the song latent representation from musical content.

It is natural to combine collaborative filtering and content models in recommendation to utilize different sources of information. A successful attempt from Wang and Blei [14], which joins a content model on article with collaborative filtering, achieves good performance on scientific article recommendation.

Inspired by these mentioned above, we create a content-aware collaborative music recommendation system. As the name suggests, the system has two components: the content model and the collaborative filtering model. To obtain a powerful content model, we pre-train a multi-layer neural network to predict semantic tags from vector-quantized acoustic feature. The output of the last hidden layer is treated as a high-level representation of the musical content, which is used as a prior for the song latent representation in collaborative filtering. We evaluate our system on the Million Song Dataset and show competitive performance to the state-of-the-art system.

2. RELATED WORK

In this section we review important relevant work. First we give an overview of matrix factorization model for recommendation, especially for *implicit feedback*. Then we describe two models which are closely related to ours: collaborative topic model for article recommendation and deep content-based music recommendation.

2.1 Recommendation by matrix factorization

A widely used approach to recommendation is collaborative filtering, where items are recommended to a user based on other users with similar patterns of item consumption. Matrix-factorization-based latent factor models [6, 8] are among the most successful collaborative filtering methods.

In a matrix factorization recommendation model, we represent both users and items in a shared low-dimensional space of dimension K, where user u is represented by a latent factor $\boldsymbol{\theta}_u \in \mathbb{R}^K$ and item i is represented by a latent factor $\boldsymbol{\beta}_i \in \mathbb{R}^K$. To make a prediction about the preference of user u on item i, we simply take the dot product between the two $\hat{r}_{ui} = \boldsymbol{\theta}_u^T \boldsymbol{\beta}_i$. To estimate user and item factors, we can minimize the squared loss between the estimated preference and actual responses $\sum_{u,i}(r_{ui} - \hat{r}_{ui})^2$, with ℓ_2 regularization on the factors to prevent overfitting. Alternating least squares (ALS) can be employed for efficient optimization. Equivalently, we can formulate a probabilistic matrix factorization model [12] with the following generative process:

- For each user u, draw user latent factor:
$$\boldsymbol{\theta}_u \sim \mathcal{N}(0, \lambda_\theta^{-1} I_K),$$
- For each item i, draw item latent factor:
$$\boldsymbol{\beta}_i \sim \mathcal{N}(0, \lambda_\beta^{-1} I_K),$$
- For each user-item pair (u, i), draw feedback:
$$r_{ui} \sim \mathcal{N}(\boldsymbol{\theta}_u^T \boldsymbol{\beta}_i, c_{ui}^{-1}),$$

and obtain the same estimates via maximum *a posteriori*. Here c_{ui} represents our confidence on the corresponding response r_{ui}, i.e., larger value of c_{ui} indicates that there is less uncertainty about the response r_{ui}, and vice versa. This is especially crucial in the case of implicit feedback (e.g., whether user u listened to song i), because of its noisy nature. Hu *et al.* [6] propose a simple heuristic for setting the values of c_{ui} for implicit feedback [1] :

$$c_{ui} = 1 + \alpha \log(1 + r_{ui}/\epsilon)$$

where α and ϵ are tunable hyperparameters. This method achieves the state-of-the-art recommendation performance in the implicit feedback case.

2.2 Collaborative topic model

Due to its content-free nature, collaborative filtering approaches can be applied in a wide range of domains. They perform well on what is called *in-matrix* predictions, i.e., recommending items that have been consumed by some users. However, this approach suffers from the well-known problem that it is unable to recommend new items that no user has consumed, or making *out-of-matrix* predictions, where content-based models are better suited. Many efforts have been made to incorporate content into collaborative filtering. Wang and Blei [14] propose the collaborative topic regression (CTR) model for scientific article recommendation, which is particularly relevant to our proposed method.

There are two components in CTR: a matrix factorization collaborative filtering model (as described in Section 2.1) and a latent Dirichlet allocation (LDA) article content model. LDA [2] is a mixed-membership model on documents. Assuming there are K topics $\Phi = \boldsymbol{\phi}_{1:K}$, each of which is a distribution over a fixed set of vocabulary, LDA treats each document as a mixture of these topics where the topic proportion $\boldsymbol{\pi}_i$ is inferred from the data. One can understand LDA as representing documents in a low-dimensional "topic" space with the topic proportion being their coordinates. With this interpretation, the generative process of CTR is as follows:

- For each user u, draw user latent factor:
$$\boldsymbol{\theta}_u \sim \mathcal{N}(0, \lambda_\theta^{-1} I_K),$$
- For each document i,
  - Draw topic proportion $\boldsymbol{\pi}_i \sim \text{Dirichlet}(\alpha)$ [2] ,
  - Draw latent factor $\boldsymbol{\beta}_i \sim \mathcal{N}(\boldsymbol{\pi}_i, \lambda_\beta^{-1} I_K)$,
- For each user-document pair (u, i), draw feedback:
$$r_{ui} \sim \mathcal{N}(\boldsymbol{\theta}_u^T \boldsymbol{\beta}_i, c_{ui}^{-1}).$$

We can see CTR differs from [6] in that CTR assumes that the item latent factor $\boldsymbol{\beta}_i$ is close to the topic proportion $\boldsymbol{\pi}_i$ but could deviate from it if necessary. This allows the user-item interaction data to "speak for itself". An attractive characteristic of CTR is its capability of making *out-of-matrix* predictions. This is done by using the topic proportion $\boldsymbol{\pi}_i$ alone as the item latent factor: $\hat{r}_{ui} = \boldsymbol{\theta}_u^T \boldsymbol{\pi}_i$, which is not possible in the traditional collaborative filtering model.

Although CTR achieves better recommendation performance than pure collaborative filtering, it does not scale well with large data. Since the model is not conditionally conjugate: the prior on $\boldsymbol{\beta}_i$ comes from a Dirichlet-distributed random variable $\boldsymbol{\pi}_i$, topic proportion $\boldsymbol{\pi}_i$ cannot be updated analytically and slower numerical optimization method is required. To address this problem, Gopalan *et al.* [5] propose the collaborative topic Poisson factorization (CTPF). This model replaces the Gaussian likelihood and Gaussian prior in CTR with Poisson likelihood and gamma prior, thus becoming conditionally conjugate with closed-form updates. Experiments on large-scale scientific article recommendation demonstrate that CTPF performs significantly better than CTR.

The main difference that sets our method apart from collaborative topic model is the content model. As a feature extractor, LDA can only produce linear factors due to its bilinear nature. On the other hand, multi-layer neural network used by in our system is capable of capturing the non-linearities in the feature space.

[1] In [6], the observational model is on the binary indicator variable $p_{ui}=\mathbb{1}\{r_{ui} > 0\}$ rather than r_{ui}, i.e., $p_{ui} \sim \mathcal{N}(\boldsymbol{\theta}_u^T \boldsymbol{\beta}_i, c_{ui}^{-1})$. However, in this paper the response r_{ui} is itself binary, indicating whether user u has listened to song i. Thus we treat r_{ui} and p_{ui} interchangeably.

[2] The generative process for words is omitted for brevity throughout the paper. Please refer to [14] for details.

2.3 Deep content-based music recommendation

Previous attempts on content-based music recommendation have achieved promising results. van den Oord *et al.* [13] utilize a neural network to map acoustic features to the song latent factors learned from the weighted matrix factorization [6]. As a result, given a new song that no one has ever listened to, a latent factor can still be predicted from the network and recommendation can be done in the same fashion as with a regular collaborative filtering model.

Our method is very similar to this approach, but we will point out two major differences:

- First, the neural network is used for different purposes. We use it as a content feature extractor, just like LDA in the collaborative topic model. The neural network in [13] maps content directly to the latent factors learned from pure collaborative filtering, and the resulting model is expected to operate similarly to collaborative filtering even when usage data is absent.

- Since the neural network is trained to map content to the latent factors learned from the weighted matrix factorization, the performance of [13] is unlikely to surpass that of the weighted matrix factorization. What we propose in this paper, on the other hand, uses content as an *addition* to the weighted matrix factorization, in a similar manner as the collaborative topic model described in Section 2.2. As we show in the experiment, we are able to achieve better result than the weighted matrix factorization when we only have limited amount of user feedback.

Other approaches that hybridize content and collaborative models include Yoshii *et al.* [17], McFee *et al.* [11], and Wang and Wang [15]. [17] train a three-way probabilistic model that joins user, item, and content by a latent "topic" variable; the model focuses on explicit feedback (user ratings). [11] take a similar approach to [13] and learn a content-based similarity function from collaborative filtering via metric learning. [15] also use a neural network to incorporate music content into the collaborative filtering model. The major difference is that in [15] the output of the neural network is treated as item factor and the neural network is trained to minimize a collaborative-filtering-based loss function. Therefore the content model itself does not have explicit musicological meaning.

3. PROPOSED APPROACH

Adopting the same structure as that of CTR, our system consists of two components: a content model which is based on a pre-trained neural network and a collaborative filtering model based on matrix factorization.

3.1 Supervised pre-training

Inspired by the success of transfer learning in computer vision which exploits deep convolutional neural networks [9], in our system we pre-train a multi-layer neural network in a supervised semantic tagging prediction task and use it as the content model.

Our training data comes from Liang *et al.* [10] which consists of $370K$ tracks from the Million Song Dataset and the pre-processed *last.fm* data with a vocabulary of 561 tags, including genre, mood, instrumentation, etc. We use the Echonest's timbre feature, which is very similar to MFCC. To get the song-level features, we vector-quantize all the timbre features following the standard procedure: We run the k-means algorithm on a subset of randomly selected training data to learn $J = 1024$ cluster centroids (codewords). Then for each song, we assign each segment (frame) to the cluster with the smallest Euclidean distance to the centroid. We aggregate the VQ feature of song i ($\boldsymbol{x}_i \in \mathbb{R}_+^J$) by counting the number of assignments to each cluster across the entire song and then normalize it to have unit ℓ_1 norm to account for the various lengths.

We treat music tagging as a binary classification problem: For each tag, we make independent predictions on whether the song is tagged with it or not. We fit the output of the network $f(\boldsymbol{x}_i) \in \mathbb{R}^{561}$ into logistic regression classifiers. Therefore, given tag labels $y_{it} \in \{-1, 1\}$ for song i and tag t, the network is trained to minimize the following loss:

$$\mathcal{L}_{\text{tag}} = \textstyle\sum_{i,t} \log(1 + \exp(-y_{it} f_t(\boldsymbol{x}_i))$$

Here we use a network with three fully-connected hidden layers and ReLU activations with dropout. Each layer has 1,200 neurons. Stochastic gradient descent with mini-batch of size 100 is used with AdaGrad [3] for adjusting the learning rate [3]. We notice that both dropout and AdaGrad are crucial for getting the good performance. The tagging performance is reported in Section 4.1.

3.2 Content-aware collaborative filtering

We can interpret the output of the last hidden layer $\boldsymbol{h}_i \in \mathbb{R}^{F_h}$ (here $F_h = 1200$) as a latent content representation of song i. Because of the way the network is trained, this latent representation is supposed to be highly correlated to the semantic tags ("topics" of music). Therefore, we can take a similar approach to the collaborative topic model and use this representation in a collaborative filtering model.

The generative process for the proposed model is as follows:

- For each user u, draw user latent factor:

$$\boldsymbol{\theta}_u \sim \mathcal{N}(0, \lambda_\theta^{-1} I_K).$$

- For each song i, draw song latent factor:

$$\boldsymbol{\beta}_i \sim \mathcal{N}(W\boldsymbol{h}_i, \lambda_\beta^{-1} I_K).$$

- For each user-song pair (u, i), draw implicit feedback (whether user u listened to song i):

$$r_{ui} \sim \mathcal{N}(\boldsymbol{\theta}_u^T \boldsymbol{\beta}_i, c_{ui}^{-1}).$$

[3] The source code for training the neural network is available at: `https://github.com/dawenl/deep_tagging`

Here the weight matrix $W \in \mathbb{R}^{K \times F_h}$ transforms the learned content representation from the neural networks into the collaborative filtering latent space via $W\boldsymbol{h}_i$. The precision parameter λ_β balances how the song latent vector $\boldsymbol{\beta}_i$ deviates from the content feature. We set the confidence c_{ui} in the same way as in Section 2.1.

We want to emphasize that our proposed model is *content-aware* instead of *content-based*. Just like collaborative topic model, our proposed model is still fundamentally based on collaborative filtering. The content model is only used as a prior and can be deviated if the model thinks it is necessary to explain the data.

For notational convenience, we define the concatenated user latent factors matrix $\Theta \stackrel{\triangle}{=} [\boldsymbol{\theta}_1|\cdots|\boldsymbol{\theta}_U] \in \mathbb{R}^{K \times U}$ and song latent factors matrix $B \stackrel{\triangle}{=} [\boldsymbol{\beta}_1|\cdots|\boldsymbol{\beta}_I] \in \mathbb{R}^{K \times I}$. We estimate the model parameters $\{\Theta, B, W\}$ via maximum *a posteriori*.

The complete log-likelihood is written as:

$$\mathcal{L} = -\sum_{u,i} \frac{c_{ui}}{2}(r_{ui} - \boldsymbol{\theta}_u^T\boldsymbol{\beta}_i)^2 - \frac{\lambda_\theta}{2}\sum_u \boldsymbol{\theta}_u^T\boldsymbol{\theta}_u - \frac{\lambda_\beta}{2}\sum_i (\boldsymbol{\beta}_i - W\boldsymbol{h}_i)^T(\boldsymbol{\beta}_i - W\boldsymbol{h}_i)$$

Take the gradient of the complete log-likelihood with respect to the model parameters and set it to 0, we can obtain the following closed-form coordinate updates:

$$\boldsymbol{\theta}_u \leftarrow (BC_uB^T + \lambda_\theta I_K)^{-1}BC_u\boldsymbol{r}_u \quad (1)$$

$$\boldsymbol{\beta}_i \leftarrow (\Theta C_i\Theta^T + \lambda_\beta I_K)^{-1}(\Theta C_i\boldsymbol{r}_i + \lambda_\beta W\boldsymbol{h}_i) \quad (2)$$

$$W^T \leftarrow (H^TH + \lambda_W I_{F_h})^{-1}H^TB^T \quad (3)$$

where $C_u \in \mathbb{R}^{I \times I}$ is a diagonal matrix with c_{ui}, $i = 1, \cdots, I$ as its diagonal elements, and $\boldsymbol{r}_u \in \mathbb{R}^I$ is the feedback for user u. C_i and $\boldsymbol{r}_i$ are similarly defined. $H \in \mathbb{R}^{I \times F_h}$ is the concatenated output from the last hidden layer $[\boldsymbol{h}_1|\cdots|\boldsymbol{h}_I]^T$. When updating W, we add a small ridge term λ_W to the diagonal of the matrix to regularize and avoid numerical problems when inverting. Alternating between updating Θ, B, and W, we are guaranteed to reach a stationary point of the complete log-likelihood.

The same technique used in [6] to speed up computation can be applied here. This enables us to apply our model to large-scale music corpus and user-item interaction, which is not possible for CTR.

After the model is trained, we can make *in-matrix* prediction by $\hat{r}_{ui} = \boldsymbol{\theta}_u^T\boldsymbol{\beta}_i$. Similar to the collaborative topic model, we can also make *out-of-matrix* prediction for songs that no one has listened to by only using the content $\hat{r}_{ui} = \boldsymbol{\theta}_u^T(W\boldsymbol{h}_i)$.

4. EVALUATION

We first evaluate our system on the pre-training tag prediction task to ensure the quality of the extracted features, and then measure its recommendation performance in comparison with related models [4].

[4] `https://github.com/dawenl/content_wmf` contains the source code for training the proposed model and reproducing the experimental results for recommendation in Section 4.2.

Model	Prec	Recall	F-score	AROC	MAP
SPMF	0.127	0.146	0.136	0.712	0.120
NNet	0.184	0.207	0.195	0.781	0.178

Table 1: Annotation and retrieval performance on the Million Song Dataset from Poisson matrix factorization with stochastic inference (SPMF) [10] and the pre-trained neural network (NNet) described in Section 3.1. The standard error is on the order of 0.01, thus not included here.

4.1 Tag prediction

Evaluation tasks and metrics We evaluate the pre-trained neural network on semantic tags with an annotation task and a retrieval task. We use the same dataset in Liang *et al.* [10] from the Million Song Dataset [1] and compare with their result which, to our knowledge, is the state-of-the-art performance on large-scale tag prediction. Note that we only use tag prediction as a proxy to measure the quality of the content model and do not argue for our approach as an optimal one to automatic music tagging.

For the annotation task we seek to automatically tag unlabeled songs. To evaluate the model's ability to annotate songs, we compute the average per-tag precision, recall, and F-score on the held-out test set. For the retrieval task, given a query tag we seek to provide a list of songs which are related to that tag. To evaluate retrieval performance, for each tag in the vocabulary we ranked each song in the test set by the predicted probability. We then calculate the area under the receiver-operator curve (AROC) and mean average precision (MAP) for each ranking.

Tagging performance and discussion The results are reported in Table 1, which show that the pre-trained neural network performs significantly better than the Poisson-factorization-based approach. This is not surprising for two reasons: 1) Here we treat tag prediction as a supervised task and train a multi-layer neural network, while in [10] the problem is formulated as an unsupervised learning task to account for the uncertainty in the user-generated tags (which incidentally can be considered as a typical example of implicit feedback). 2) Similar to LDA, Poisson factorization can only capture linear factor, whose expressive power is much weaker than that of a multi-layer neural network.

Nevertheless, the results confirm that our pre-trained neural network can be considered as an effective content feature extractor and we will use the output of the last hidden layer as the content feature.

Note that our neural network is relatively simple and does not directly use raw acoustic features (e.g., log-mel spectrograms) as input. It is reasonable to believe that with a more complex network structure and low-level acoustic feature, we should be able to achieve better tagging performance and obtain a more powerful content feature extractor, which could further boost the performance of our proposed recommendation method.

Model	$R@40$	$R@80$	$R@120$	$R@160$	$R@200$	NDCG
PMF [4]	0.1021	0.1533	0.1908	0.2206	0.2456	0.2419
CTPF [5]	0.1031	0.1511	0.1861	0.2138	0.2370	0.2395
WMF [6]	0.1722	0.2367	0.2803	0.3133	0.3397	0.2881
CF + shallow	0.1724	0.2368	0.2803	0.3131	0.3396	0.2883
CF + deep	0.1722	0.2365	0.2800	0.3129	0.3394	0.2882

Table 2: *In-matrix* performance on the DEN subset with proposed and competing methods.

4.2 Recommendation

Data preparation We use the Taste Profile Dataset which is part of the Million Song Dataset to evaluate the recommendation performance. It contains listening history in the form of play counts from one million users with more than 40 million (user, song, play count) triplets. We first binarize all the play counts [5] and create two complementary subsets (denoted as *DEN* and *SPR*):

For the DEN subset, we intend to create a reasonably dense subset so that the traditional collaborative filtering model will have good performance. We remove the users who have less than 20 songs in their listening history and songs that are listened to by less than 50 users, obtaining a subset with 613,682 users and 97,414 songs with more than 38 million user-song pairs (sparsity level 0.064%). For the SPR subset, on the contrary, we only keep the users who have less than 20 songs in their listening history and songs that are listened to by less than 50 users, yielding a highly sparse (0.002%) subset with 564,437 users and 260,345 songs.

We select 5% of the songs from DEN (4,871) for *out-of-matrix* prediction. For both subsets we split 20% and 10% as test and validation sets, respectively. Validation set is used to select hyperparameters, as well as monitor convergence by computing predictive likelihood.

Competing methods We compare our proposed method (denoted as *CF + deep*) with weighted matrix factorization (*WMF*) [6], as well as the following three methods:

CF + shallow: A simple baseline where we directly use the normalized VQ feature $\boldsymbol{x}_i$ in place of the feature extracted from the neural network $\boldsymbol{h}_i$. This baseline is mainly used to demonstrate the necessity of an effective feature extractor for *out-of-matrix* prediction.

Poisson matrix factorization (*PMF*) [4]: Just like WMF, PMF is a matrix factorization model for collaborative filtering. Instead of Gaussian likelihood and priors on the latent factors, it utilizes Poisson likelihood model and gamma priors. The biggest advantage of PMF is computational. As shown in [4], the inference algorithm has complexity that scales linearly with the number of non-zero entries in the user-item matrix.

Collaborative topic Poisson factorization (*CTPF*) [5]: This model incorporates the content information into PMF in the same way as CTR. Additionally, it is conditionally conjugate with closed-form updates and enjoys the same computational efficiency as PMF. Therefore, it can be applied to large-scale dataset without delicate engineering.

Based on our argument in Section 2.3, we do not directly compare with [13] because it is sufficient to compare with WMF. For *out-of-matrix* recommendation evaluation, we can only compare with CTPF and CF + shallow. In all the experiments, the dimensionality of the latent space $K = 50$. We select $\alpha = 2$ and $\epsilon = 10^{-6}$ to compute the confidence c_{ui}. For WMF, CF + shallow, and CF + deep, the model parameters Θ, B and W (if any) are initialized to the same values.

Evaluation metrics To evaluate different algorithms, we produce a ranked list of all the songs (excluding those in the training and validation sets) for each user based on the predicted preference $\hat{\boldsymbol{r}}_u$.

Precision and recall are commonly used evaluation metrics. However, for implicit feedback, the zeros can mean either the user is not interested in the song or more likely, the user does not know the song. This makes the precision less interpretable. However, since the non-zero r_{ui}'s are known to be true positive, we instead report $Recall@M$, which only considers songs within the top M in the ranked list. For each user, the definition of $Recall@M$ is

$$Recall@M = \frac{\text{\# songs that the user listened to in top } M}{\text{total \# songs the user has listened to}}.$$

In addition to $Recall@M$, we also report (untruncated) normalized discounted cumulative gain (NDCG) [7]. Unlike $Recall@M$ which only focuses on top M songs in the predicted list, NDCG measures the global quality of recommendation. In the meantime, it also prefers algorithms that place held-out test items higher in the list by applying a discounted weight. Given a ranked list of songs from the recommendation algorithm, for each user NDCG can be computed as follows:

$$\text{DCG} = \sum_{i=1}^{I} \frac{2^{rel_i} - 1}{\log_2(i+1)}; \quad \text{NDCG} = \frac{\text{DCG}}{\text{IDCG}}.$$

Given our binarized data, the reverence rel_i is also binary: 1 if song i is in the held-out user listening history and 0 otherwise. IDCG is the optimal DCG score where all the held-out test songs are ranked top in the list. Therefore, larger NDCG values indicate better performance.

Results on the DEN subset The model hyperparameters $\lambda_\theta = \lambda_W = 10$ and $\lambda_\beta = 100$ are selected from the validation set based on NDCG. The *in-matrix* and *out-of-matrix* performances are reported in Table 2 and 3, respectively.

[5] In practice, we find that the performances using actual play counts and binarized indicators are very close for our model.

Model	$R@40$	$R@80$	$R@120$	$R@160$	$R@200$	NDCG
CTPF [5]	0.0256	0.0700	0.1440	0.1869	0.2086	0.1271
CF + shallow	0.0503	0.0894	0.1218	0.1514	0.1778	0.1429
CF + deep	**0.0910**	**0.1461**	**0.1881**	**0.2241**	**0.2550**	**0.1605**

Table 3: *Out-of-matrix* performance on the DEN subset with proposed and competing methods.

Model	$R@40$	$R@80$	$R@120$	$R@160$	$R@200$	NDCG
WMF [6]	0.1137	0.1286	0.1378	0.1449	0.1505	0.1415
CF + shallow	0.1138	0.1286	0.1377	0.1449	0.1504	0.1416
CF + deep	**0.1140**	**0.1289**	**0.1378**	**0.1451**	**0.1507**	**0.1417**

Table 4: *In-matrix* performance on the SPR subset with proposed and competing methods.

All the metrics are averaged across 612,232 users in the held-out test user-item pairs.

We can see that with sufficient amount of user feedback, there is almost no difference in performance among WMF, CF + shallow, and CF + deep [6] – there is not a single model which is consistently better. This is understandable, since both CF + shallow and CF + deep are fundamentally collaborative filtering models. With enough user feedback, the model is able to produce meaningful recommendation without resorting to the content features. Moreover, CF + shallow, which has access to more content information, does slightly better than CF + deep.

One observation from Table 2 is that adding content features does not necessarily improvement the performance. Unlike CF + deep, CTPF falls behind its content-free counterpart PMF on both $Recall@M$ and NDCG. This is possibly due to the insufficient feature extraction capability of the topic model (LDA) on the rich musical data.

The superiority of CF + deep is more obvious on the *out-of-matrix* predictions performance shown in Table 3. We can see a larger margin between CF + deep and CF + shallow, as compared to their close performance on *in-matrix* predictions. This suggests the importance of a powerful feature extractor in the absence of usage data. Even a simple linear LDA model in CTPF can be more effective than CF + shallow at predicting songs that the users listened to in the held-out test set.

Results on the SPR subset We repeat the *in-matrix* evaluation on the highly sparse SPR subset. The model hyperparameters $\lambda_\theta = \lambda_W = 10^{-2}$ and $\lambda_\beta = 1$ are selected from the validation set. The performance is reported in Table 4. All the metrics are averaged across 564,437 users in the held-out test user-item pairs.

Again, the overall differences among all three methods are relatively minor. However, with very limited user feedback, both CF + shallow and CF + deep outperform the content-free WMF. More importantly, CF + deep consistently improves over CF + shallow, which indicates the importance of an effective feature extractor.

[6] There is little point in arguing for the statistical significance of the difference, since given the number of users to average over, the standard error is vanishingly small.

5. CONCLUSION

In this paper we present a content-aware collaborative music recommendation system that joins a multi-layer neural network content model with a collaborative filtering model. The system achieves the state-of-the-art performance in music recommendation given content and implicit feedback data.

A possible future direction is to incorporate ranking-based loss function, e.g., the weighted approximate-rank pairwise (WARP) loss in [16] into the collaborative filtering model. We normally evaluate recommendation algorithms using ranking-based metrics (e.g. $Recall@M$ and NDCG), but the model is trained using squared loss function. It would be more natural to directly optimize a ranking-based loss function.

6. ACKNOWLEDGEMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research.

7. REFERENCES

[1] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In *ISMIR*, 2011.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[3] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

[4] Prem Gopalan, Jake M. Hofman, and David M. Blei. Scalable recommendation with poisson factorization. *arXiv preprint arXiv:1311.1704*, 2013.

[5] Prem K. Gopalan, Laurent Charlin, and David Blei. Content-based recommendations with Poisson factorization. In *Advances in Neural Information Processing Systems 27*, pages 3176–3184. 2014.

[6] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. IEEE, 2008.

[7] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[8] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[10] Dawen Liang, John Paisley, and Daniel P. W. Ellis. Codebook-based scalable music tagging with Poisson matrix factorization. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 167–172, 2014.

[11] Brian McFee, Luke Barrington, and Gert Lanckriet. Learning similarity from collaborative filters. In *ISMIR*, pages 345–350, 2010.

[12] Andriy Mnih and Ruslan Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2007.

[13] Aäron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*, pages 2643–2651, 2013.

[14] Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 448–456. ACM, 2011.

[15] Xinxi Wang and Ye Wang. Improving content-based and hybrid music recommendation using deep learning. In *Proceedings of the 22nd ACM International Conference on Multimedia*. ACM Press, 2014.

[16] Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI*, 2011.

[17] Kazuyoshi Yoshii, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. In *ISMIR*, 2006.