Recommending Music Curators: A Neural Style-Aware Approach

Jianling Wang(✉) and James Caverlee

Texas A&M University, College Station, TX, USA
{jlwang,caverlee}@tamu.edu

Abstract. We propose a framework for personalized music curator recommendation to connect users with curators who have matching *curation style*. Three unique features of the proposed framework are: (i) models of curation style to capture the coverage of music and curator's individual style in assigning tracks to playlists; (ii) a curation-based embedding approach to capture inter-track agreement, beyond the audio features, resulting in models of music tracks that pair well together; and (iii) a novel neural pairwise ranking model for personalized music curator recommendation that naturally incorporates both curator style models and track embeddings. Experiments over a Spotify dataset show significant improvements in precision, recall, and F1 versus state-of-the-art.

1 Introduction

Music streaming platforms provide access to a diverse, incredibly large, and ever growing collection of music tracks. To make sense of the millions of available tracks (e.g., 40 million on Apple Music and 30 million on Spotify), playlists have become an essential feature of many music streaming platforms for organizing music, mediating how users experience the service. Across platforms, most playlists are manually curated and managed by a group of *music curators*, which consists of both "regular" users and expert tastemakers. To benefit from the power of human curation [15], many platforms enable users to follow these music curators to receive updates of their listening activities, e.g., to discover new tracks, albums, or playlists (as illustrated in Fig. 1(a)).

While recommendation systems have been widely deployed in many music streaming platforms for tasks like recommending individual music tracks [3,19,23] or playlists [2,14], they are not well-suited for real-world scenarios like (i) discovering new tracks with little or no feedback; (ii) finding relevant playlists that are frequently updated (and hence, out-of-sync with respect to a learned recommendation model); and (iii) recommending playlist creators themselves who can provide direct access to new tracks, albums, or playlists. As a step toward supporting these scenarios, we focus on the task of *curator recommendation* to create a personalization layer to help users discover vast amounts of new tracks, fresh playlists, and interesting curators.

While some services highlighting highly-rated or popular curators [7,16,22] (e.g., Spotify's recommendation of "featured" curators with high popularity),

J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12035, pp. 191–204, 2020.
https://doi.org/10.1007/978-3-030-45439-5_13

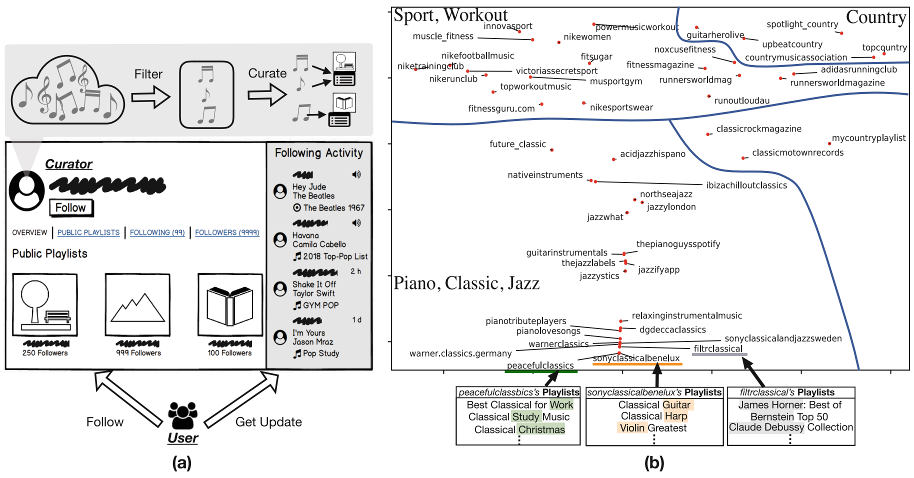

Fig. 1. (a) Users follow *music curators* with matching curation style to receive updates. (b) We randomly sample curators with IDs containing the keywords ("workout", "sport", "run", "fit", "gym", "country", "piano", "instrument", "classic", "jazz"). We show the 2D visualization (with t-SNE [11]) of selected curators based on average audio features of music tracks they curate.

identifying personally-relevant music curators is a daunting task due to the following challenges. First, music curators themselves are complex amalgamations of the playlists they create, the tracks they select, and their unique *style*. For example, some curators may focus on specific emotions (like happy or excited), eras (like the 80s or 90s), or situations (like workouts, parties, or road trips), while others cross boundaries (like happy 80s music, or 90s road trip). Hence the first challenge is: How can we build models that capture these stylistic differences across music curators taking both their curating coverage and individual style into consideration? To illustrate, we conduct an initial exploration of Spotify curators whose areas of interest can be inferred from their user IDs. We represent each of the music curators using the average of audio features (see Sect. 3.2 for details) for all the tracks in playlists they curate and plot the 2D t-SNE distribution in Fig. 1(b). We see that there are clear patterns of curator *coverage*: country music curators cluster together in the top-right, curators focusing on active music for sports and workouts cluster in the top-left, while classical and instrumental music curators dominate the lower portion. We see that *curators have preferences in the coverage of music* they would curate. However, coverage alone is insufficient to distinguish between curators. We must also consider individual *style*. Considering the playlists curated by three classical music curators (in the bottom of the figure), while all drawing from the same musical coverage area, each of them displays a unique style – one groups tracks for activities like work or study, one collects tracks featuring the same instrument, while the third groups by artists. We see that *curators who curate similar types of music can have different style in deciding how tracks go with each other.*

Second, user preferences for music curators may be driven by many factors, including preferences for curator coverage and style. These preferences may only be revealed through extremely sparse user feedback. For example, in a sample of Spotify playlists (see Sect. 4.1) we find that only 0.20% of all user-curator pairs have a following relationship. And for pairs without a following relationship, it may mean the user dislikes the curator or just has not known her yet. Furthermore, because anyone can be a curator on these platforms, there are many long-tail candidate curators who may be invisible to most users. Hence the second challenge is: How can we uncover the hidden taste preferences that connect users to the music curators they may prefer?

Toward answering these questions, we propose a novel personalized **M**usic **C**urator **R**anking (MCR) framework to recommend music curators to users based on the style of each curator and on each user's taste profile. There are three unique features of the proposed MCR framework: (i) We propose to model *curation style* through a novel neural pairwise framework that considers how each curator assigns tracks to different playlists, toward uncovering each curator's latent style; (ii) Based on how crowds of curators compose their playlists, we propose an embedding model for tracks to capture inter-track agreement to uncover hidden connections among tracks, which can assist in characterizing the coverage of each curator; (iii) We propose a novel neural ranking model for personalized music curator recommendation that naturally incorporates both curator style models and track embeddings to identify personally relevant curators. Through experiments over a Spotify dataset, we find the proposed framework results in a 20.5% and 5.7% improvement in top-k F1 score compared to Bayesian Personalized Ranking (BPR) and Neural Personalized Ranking (NPR), and in a 24.9% and 21.4% improvement in cold start scenarios.

2 Related Work

Music Recommendation and Playlist Generation. Complementary to our focus on recommending curators, many researchers have explored music continuation and automated playlist generation. For example, [12] predicts the next track based on a listener's preferences and most-recently played tracks. DJ-MC [14] aims to recommend track sequences based on reinforcement learning. Groove Radio [1] generates personalized playlists based on seed artists. EFM [2] recommends both tracks and playlists through a new embedding approach. There are also efforts on training an embedding model on users' historical music playing sequences to estimate the similarity between songs for recommendation [5,24]. In contrast, we propose a *style-based* recommender that links users directly to curators rather than specific tracks or playlists.

User Profiling and Expert Recommendation. Somewhat similar to our notion of curator is research on finding expertise to improve search and recommendation [9,27]. By uncovering the latent preferences of users or building up profiles for them, these approaches aim to identify related experts. For music, one effort has aimed to identify curators [13] based on a Linked Data graph capturing listening history and other factors. However, it cannot scale to larger datasets demonstrating sparsity as in our case.

Neural-Based Recommendation with Implicit Feedback. To make recommendations in such sparse, implicit feedback scenarios, methods like Bayesian Personalized Recommendation (BPR) [21] and a recently introduced variant called Neural Personalized Ranking (NPR) [18] have shown good success. These and other neural approaches have demonstrated their power in recommenders, including [4,6,25]. Other approaches include the autoencoder-based CDAE [26] and Neural Collaborative Filtering [10] that adds nonlinearities to traditional Matrix Factorization (MF). In this work, we propose to take advantage of the benefits of neural architectures for recommenders, while carefully incorporating special properties of music curation, including curator style, coverage, and curation-based embeddings.

3 MCR: Music Curator Ranking

In this section, we start from problem setting and then present the design of our MCR framework, organized around three guiding research questions.

Problem Setting: Let $U = \{u_1, u_2, ..., u_N\}$ be a set of N users and $C = \{c_1, c_2, ..., c_M\}$ be a set of M curators. A curator c can create a playlist L_q^c composed of tracks drawn from a collection of possible tracks $T = \{t_1, t_2, ..., t_{|T|}\}$. Further, curator c can create multiple playlists $P_c = \{L_1^c, L_2^c, ..., L_Q^c\}$. Users may express their interest in a curator through an action such as a "like" or "follow". Our goal is to recommend a personalized ranked list of music curators to each user.

Research Question: RQ1: How to model the hidden *curation style* that guides how playlist curators select tracks for their playlists? **RQ2:** How can we use these curation decisions to model individual tracks, to capture how curators view tracks beyond their particular audio characteristics? **RQ3:** How to model users preferences on both curation style and coverage for improved recommendation?

3.1 RQ1: Model of Curation Style

As shown in the previous section, curators have their own style in deciding how tracks go together. For example, some classical music lovers may create playlists based on the time period (e.g., 1800 s) or for different artists. Some may focus on the particular instrument featured, while others may curate based on feelings, activities (e.g., for studying or focusing), or locations (e.g., for "coffee shop"). Modeling these styles is important for accurately connecting users to their preferred curators. A purely content-based approach to model style (e.g., based on composer, time period, or audio features of the track) may face challenges in determining these subtle stylistic choices that motivate a curator.

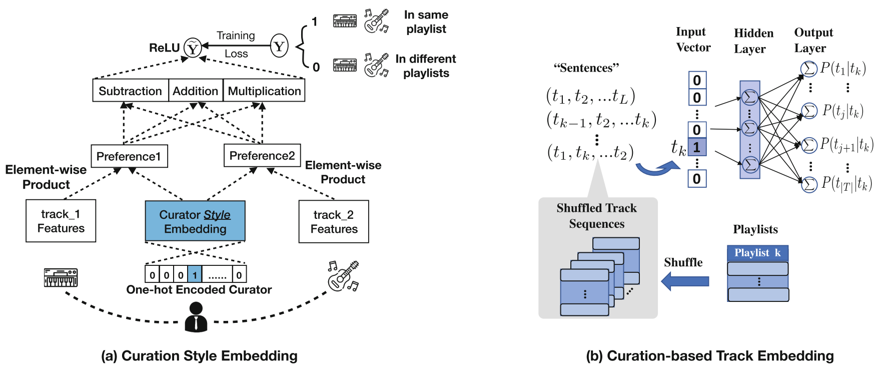

Fig. 2. Curation style modeling and curation-based track embedding.

Hence, we propose to model *curation style* through a pairwise framework intended to discern why a curator chooses one track over another. Given tracks t_1, t_2, and t_3, curator c may put t_1 and t_2 in the same list while putting t_3 in a different list. We propose a neural network (see Fig. 2(a)) to simulate how curators choose what tracks go in what playlists. By iterating among and across playlists of different curators, we can generate input tuples to represent their behaviors in assigning tracks to playlists. Each input tuple has three components: the index of the curator, the features representing t_1, and the features representing t_2. If curator c puts t_1 and t_2 into the same playlist (a positive pair), then the ground truth of prediction is set to be 1. However, if t_1 and t_2 appear in different playlists of curator c (a negative pair), then the ground truth is set to be 0. We can then train a binary classification model to predict how curators will relate two tracks during curation. The intermediate embedding layer of this model – $\mathbf{e}_c^{style}$ – can then characterize curator c's curation style.

Concretely, such a model requires a vector representation as input. Let $\mathbf{i}_c$ denote the one-hot encoding of curator c. Since the space of tracks is large, instead of a one-hot approach, we represent each track t_i by a dense vector $\mathbf{F}_{t_i}$ (more details in Sect. 3.2). After feeding the input tuple $< c, \mathbf{F}_{t_1}, \mathbf{F}_{t_2} >$ to the neural network, to learn the representation of c's curation style, the vector $\mathbf{i}_c$ is passed to the embedding layer: $\mathbf{e}_c^{style} = \mathbf{W}_{style}\mathbf{i}_c$, where $\mathbf{W}_{style}$ is the embedding matrix and the resulting embedding $\mathbf{e}_c^{style}$ will be used to characterize c. Then the element-wise multiplication between embedding of the curator and music track is analogous to matrix factorization with $\mathbf{H}_c^{t_1} = \mathbf{e}_c^{style} \circ \mathbf{F}_{t_1}$ and $\mathbf{H}_c^{t_2} = \mathbf{e}_c^{style} \circ \mathbf{F}_{t_2}$. Thus the resulting vectors $\mathbf{H}_c^{t_1}$ and $\mathbf{H}_c^{t_2}$ represent c's preferences in curating. To decide whether the curator will put t_1 and t_2 into the same or different playlists, we calculate the element-wise absolute difference between them $|\mathbf{H}_c^{t_1} - \mathbf{H}_c^{t_2}|$ for comparison. We also calculate the element-wise multiplication $\mathbf{H}_c^{t_1} \circ \mathbf{H}_c^{t_2}$ to represent the cosine distance between them. Then the concatenated vector will be fully connected to a dense layer activated with ReLU function. Finally a one-dimension output is generated as the prediction of likelihood that t_1 and t_2 will be in the same playlist by curator c. Binary cross-entropy is used to calculate the training loss. After reaching an accurate prediction model, the embedding vector $\mathbf{e}_c^{style}$ can be used to represent c's curation style.

Curator Coverage. Complementary to curation style, we can also characterize a curator's coverage as a simple aggregate of the tracks associated with a curator. Let $T_c = \{t_i | t_i \in L_x^c, \forall L_x^c \in P_c\}$ represent the set of tracks that curator c adds to at least one playlist, where P_c denotes the set of playlists curated by c and L_x^c is each of those playlists. Recall that each track t_i can be represented by a dense vector $\mathbf{F}_{t_i}$. Concretely, we can use the weighted sum of T_c to model c's curating coverage, in which the weight of each track is the frequency with which the track appears across all of curator c's playlists. Then we can describe the range of musical style curator c covers using the weighted average of set T_c:

$$\mathbf{cov}_c = \frac{1}{\sum_{t_i \in T_c} count_c(t_i)} \sum_{t_i \in T_c} count_c(t_i)\mathbf{F}_{t_i}, \tag{1}$$

where $count_c(\cdot)$ counts the frequency of appearing across all the playlists of c.

3.2 RQ2: Curation-Based Track Embeddings

A natural question raised in the last section is how to model individual tracks in the first place? One alternative is to use the *audio features* of music tracks. For example, there are 13 audio features provided by Spotify, including *danceability, energy, key, loudness, mode, speechless, acousticness, instrumentalness, liveness, valence, tempo, duration in ms*, and *time signature*. However, these features are not always available due to the cost of analyzing audio signals.

Instead, we seek to characterize tracks by how groups of curators compose their playlists. The intuition is that when users create or follow playlists, they provide implicit linkages between tracks in the same playlist. Hence, a key hypothesis is that tracks in a particular playlist are in high coherence, regardless of their underlying audio-based signature. Inspired by word2vec [17], we propose to learn a vector representation for each track from the curation-based perspective of how tracks cohere with other tracks. We can treat tracks as "words" and find neighboring tracks within a window. In many music services, users can shuffle the playlists, meaning that tracks arrive in a random order. Hence, tracks within a playlist will have high coherence, regardless of their immediate order in the playlist. To simulate this shuffle activity, we randomly reorder the music tracks in each playlist to generate the shuffled track sequences (see Fig. 2(b)). Then we treat each shuffled track sequence as a "sentence" and each music track as a "word". For instance, given a music sequence (similar to a "sentence") $\boldsymbol{m} = (t_1, t_2, ...t_{|\boldsymbol{m}|})$, the log probability l_m is calculated as $l_{\boldsymbol{m}} = \frac{1}{|\boldsymbol{m}|} \sum_{0 \le k < |\boldsymbol{m}|} \sum_{-w \le j \le w, j \ne 0} \log P(t_{k+j}|t_k)$, where $P(t_{k+j}|t_k)$ represents the probability that track t_{k+j} is the neighbor of t_k, given that track t_k is listened. Here, w is the window size in observing neighboring tracks. This skip-gram model aims to maximize the log probability across the set of all generated track "sentences". The hidden layer (in Fig. 2(b)) that is learned will be used as the low-dimensional vector representations for tracks. Tracks with larger pairwise-similarity are more likely to be listened to together from the perspective of how curators create their playlists. After generating the embedding representation $\mathbf{E}_t$ for each track t, we can use it as track features for *curation style embedding* in Fig. 2(a) and *coverage* in Eq. 1.

3.3 RQ3: Neural Personalized Curator Ranking

Given these models of curation style and embedding of individual tracks, we now turn to the challenge of connecting users with the right music curators. The main insight of the proposed MCR approach is that when deciding between two curators, a user will consider the style and coverage of each curator.

Users leave only implicit feedback on curators in the form of "following". That is, we can assume that if user u follows curator c, then u is interested in c. However, if u does not follow curator c, we cannot conclude that u is not interested in c because it is also possible that u is unaware of c. Hence, to overcome this implicit feedback challenge, we propose a neural pairwise ranking model inspired by Bayesian Personalized Recommendation (BPR) [21] and Neural Personalized Ranking (NPR) [18]. Following BPR and NPR, the key assumption of this proposed approach is that users prefer the observed positive items (following a curator) to the unobserved items.

The proposed MCR model consists of two symmetric branches as shown in Fig. 3. Suppose that user u has already followed curator c and hasn't followed curator q yet. We denote this relationship as $c >_u q$. Given user u and a pair of curators (c, q), the left branch is designed to estimate a user's overall preference for curator c, while the right branch (the transparent part) aims to estimate a user's overall preference for curator q.

There are 7 inputs to the symmetric structure, denoted as $(u, c, q, \mathbf{cov}_c, \mathbf{cov}_q, \mathbf{e}_c^{style}, \mathbf{e}_q^{style})$. Besides the index of the user u, the curator c and another curator q, we also feed in the coverage $\mathbf{cov}_c$, $\mathbf{cov}_q$ and style $\mathbf{e}_c^{style}$, $\mathbf{e}_q^{style}$ of curator c and q. While constructing the input tuple for model training, we select a curator followed by u and another curator for whom u does not leave feedback. Given the tuple (u, c, q), its ground truth label $y(u, c, q)$ is equal to 1 if $c >_u q$, while $y(u, c, q) = -1$ if $q >_u c$. Then the personalized ranking problem is transformed into a binary classification problem.

Model Details. Since MCR is symmetric, we focus on one of the branches in detail. To fit into the neural structure, the index of u and c are one-hot encoded as $\mathbf{i}_u$ and $\mathbf{i}_c$ directly after input. First, we model the direct preference relationships between users and curators. $\mathbf{i}_u$ and $\mathbf{i}_c$ are connected to the corresponding embedding layers to learn the compact and vectorized representations. The resulting embeddings $\mathbf{e}_u$ and $\mathbf{e}_c$ act as the latent factors of u and c. To simulate the traditional matrix factorization, we calculate the element-wise product $\mathbf{e}_u \circ \mathbf{e}_c$, which captures the interaction between u and c.

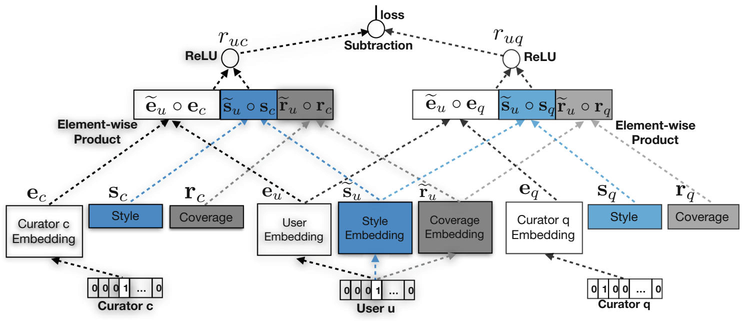
loss
Subtraction
r_{uc}
ReLU
r_{uq}
ReLU
$\widetilde{\mathbf{e}}_u \circ \mathbf{e}_c$
$\widetilde{\mathbf{s}}_u \circ \mathbf{s}_c$
$\widetilde{\mathbf{r}}_u \circ \mathbf{r}_c$
$\widetilde{\mathbf{e}}_u \circ \mathbf{e}_q$
$\widetilde{\mathbf{s}}_u \circ \mathbf{s}_q$
$\widetilde{\mathbf{r}}_u \circ \mathbf{r}_q$
Element-wise Product
Element-wise Product
$\mathbf{e}_c$
$\mathbf{s}_c$
$\mathbf{r}_c$
$\widetilde{\mathbf{e}}_u$
$\widetilde{\mathbf{s}}_u$
$\widetilde{\mathbf{r}}_u$
$\mathbf{e}_q$
$\mathbf{s}_q$
$\mathbf{r}_q$
Curator c Embedding
Style
Coverage
User Embedding
Style Embedding
Coverage Embedding
Curator q Embedding
Style
Coverage
0 0 0 1 ... 0
Curator c
0 0 0 1 ... 0
User u
0 1 0 0 ... 0
Curator q

Fig. 3. Neural Music Curator Ranking (MCR) structure.

Then we need to determine how a user's preferences are aligned with each curator's coverage and style. We use ReLU function to activate curator c's auxiliary features. The activated coverage and style of curator c are denoted as $\mathbf{r}_c$ and $\mathbf{s}_c$. Given the one-hot encoding of user u as $\mathbf{i}_u$, u's preferences on coverage and style are learned by the corresponding embedding layers $\widetilde{\mathbf{r}}_u = \mathbf{W}_r\mathbf{i}_u$ and $\widetilde{\mathbf{s}}_u = \mathbf{W}_s\mathbf{i}_u$, where $\mathbf{W}_r$ and $\mathbf{W}_s$ are the embedding weight matrices for coverage and style. The resulting embeddings share the same size with the coverage and style features of curators. We use the element-wise multiplications to capture user u's preferences on curator c. We will concatenate all the element-wise products together to represent the preference of u on c and get $\mathbf{p}_{uc} = [\mathbf{e}_u \circ \mathbf{e}_c \quad \widetilde{\mathbf{r}}_u \circ \mathbf{r}_c \quad \widetilde{\mathbf{s}}_u \circ \mathbf{s}_c]^T$. Then this concatenation is passed to a fully connected layer. The one-dimension output from this layer can be represented as $r_{uc} = f(\mathbf{W}\mathbf{p}_{uc}+b)$. Here $f(\cdot)$ denotes the ReLU function. $\mathbf{W}$ is the weight matrix and b is the bias term for the single perceptron. The output r_{uc} represents the preference score of u on c. Because the model is symmetric, r_{uq}, the preference score of u on curator q, is calculated in the same process with the other branch (the transparent part in Fig. 3). Thus we apply a subtraction layer to estimate the difference between u's preferences on curator c and q: $d(u,c,q) = r_{uc} - r_{uq}$. Since our objective is to make $d(u,c,q)$ share the same sign as $y(u,c,q)$, we train the entire model to minimize the loss: $L = \frac{1}{|\boldsymbol{V}|}\sum_{(u,c,q)\in \boldsymbol{V}} -\ln(\delta(d(u,c,q)\cdot y(u,c,q)))$, in which $\boldsymbol{V}$ is the set of training tuples and δ is the sigmoid function. If the product $d(u,c,q)\cdot y(u,c,q)$ is larger, it means that u's preference for the curator pair $(c,\ q)$ is more likely to be predicted correctly by the model. We adopt the L2-norm for regularization.

During prediction, we can input the tuple $(u,\ c,\ c,\ \mathbf{cov}_c,\ \mathbf{cov}_c,\ \mathbf{e}_c^{style},\ \mathbf{e}_c^{style})$ to calculate user u's preference on curator c. We denote the preference scores generated from the symmetric components as r_{uc}^{left} and r_{uc}^{right}. Then the overall preference score can be predicted as $r_{uc} = \frac{1}{2}(r_{uc}^{left} + r_{uc}^{right})$. Thus for prediction, we only need to replace the last subtraction layer with an addition layer. With the preference score of a user over each curator, we will be able to generate a personalized ranked list of curators for this user.

4 Experiments

In this section, we conduct a series of experiments to evaluate the proposed framework by answering these questions: (i) How does the proposed MCR perform in curator recommendation compared with state-of-the-art? (ii) How does each component of MCR contribute to the quality of curator recommendation? and (iii) Are the proposed *style* features able to alleviate the cold start issue?

4.1 Setup

Data. Our experiments are based on a dataset sampled from Spotify. Initially, we create a seed list of playlists by issuing 200 keyword queries representing popular topics on Spotify (e.g., pop, coffee, trip) and then randomly selecting 0.2 million returned playlists. We then identify the creators of these playlists and crawl their followees, arriving at a list of 5 million valid user IDs. We then identify active users who have followed more than 5 music curators. We end up with a dataset with 19,760 users and 6,821 music curators who have curated 5,413,478 music tracks in total. Further, the resulting dataset is extremely sparse, in which only 0.20% of the user-curator pairs have a following relationship. We use 60% of user-curator following data for training, 10% for validation and the remaining 30% for testing.

Metrics and Baselines. We adopt Precision@k (Prec@k), Recall@k (Rec@k) and F-1 score@k (F1@k) as metrics to evaluate the personalized recommendation. Prec@k represents the percentage of correctly predicted curators among the top-k recommendations, and Rec@k represents the fraction of relevant curators which are discovered by the top-k recommendations. F1@k is a weighted combination of Prec@k and Rec@k, that is $F1@k = \frac{2 \cdot Prec@k \cdot Rec@k}{Prec@k + Rec@k}$.

We first consider 3 classic recommendation models and a graph-based model for curator recommendation as baselines:

- *Popular (MP)* recommends the curators with highest popularity.
- *User-based Collaborative Filtering (UCF)* estimates user u's preference on curator c with a weighted aggregation of his/her neighboring users' feedback.
- *Bayesian Personalized Ranking (BPR)* [21] is a basic pairwise ranking model with Matrix Factorization.
- *node2vec* [8] explores diverse neighborhoods on a graph through a biased random walk. We use *node2vec* to find embeddings for users and curators with the user-user (curators) following graph. Based on the cosine similarities between each user and all the curators, we can recommend nearby curators.

We also want to compare MCR with its simplified variants:

- *Neural Personalized Ranking (NPR)* [18] ignores both *style* and *coverage*. Recommendation is purely based on user-curator following relationships.
- *Coverage-based MCR (C-MCR)* relies only on the *coverage* features based on track embeddings and ignores *style*.

– *Audio feature-based MCR (A-MCR).* Let $\mathbf{A}_{t_i}$ denote the 13-dimension audio feature (details in Sect. 3.2) vector of music track t_i. To compare our curation-based track embeddings with the traditional audio-based features, we replace $\mathbf{F}_{t_i}$ in Eq. 1 with track embedding $\mathbf{A}_{t_i}$. This model ignores the curator *style* features.
– *Style-based MCR (S-MCR)* integrates the *style* embedding $\mathbf{e}^{style}$ as a contextual feature for MCR, while ignoring the simpler *coverage* feature.

Parameter Settings. For the curator *style* embedding, we randomly pick 10 tracks from each playlist and iterate among all playlists to get the positive tuples. Then for each curator, we iterate across playlists he/she creates to generate negative tuples. We keep the number of positive tuples and the number of negative tuples to be the same for each curator by random sampling. With this method, we generate a training set and a validation set with the same size. We train the model for 20 epochs, with the loss generally converging within 10 epochs. We use the intermediate embedding layer in the well-trained model to represent a curator's *style*. For the curation-based track embeddings, we set the window and feature vector size following the default settings of Gensim [20].

We adopt Adam optimizer and mini-batch approach of gradient descent, in which the batch size is 3072. We grid search the regularization parameters over $\{1, 10^{-1}, ...10^{-6}\}$ in the validation set. We set the dimension of latent factors for "following" feedback to be 100 for all the models. The dimensions of *coverage* based on track embeddings and curation style embedding $\mathbf{e}^{style}$ are also set to be 100. We keep them to be the same for fair comparison. To train the pairwise ranking model, we select 5 negative curators (unfollow/unobserved) for each user to construct the training tuples; we find that increasing these negative samples leads to longer training time but little improvement on recommendation quality.

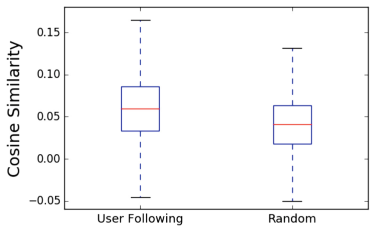

Fig. 4. Similarity of users and curators on curation style.

Exploring Curator Style. Firstly, we want to explore what the *style* embeddings discover about curators. Although users may not create playlists themselves, they can reveal their "curation style" by following different playlists. For a user u, we calculate the cosine similarity of the curation style embedding between u and two different sets of curators: (i) we select the set of K curators u is *following*; and (ii) we *randomly* select K curators. We summarize the results for all the users in Fig. 4. We observe that *users have a higher similarity with curators they follow rather than random curators.* Thus, we see that users do have clear preferences based on style, and there exists clear pattern of users sharing style with curators.

4.2 Evaluating Curator Recommenders

In our first experiment, we compare MCR versus the baselines. We report the precision, recall and F1 over the testing set in Fig. 5. We see that the proposed MCR approach results in the highest precision, recall and F1 for k = 5, 10, 15.

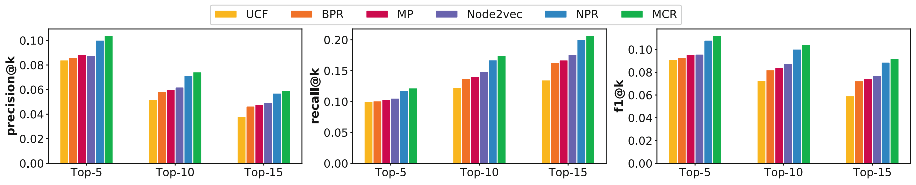

Fig. 5. Comparing models: top-k precision, recall and F1.

Beginning on the left-side of each figure, we see that user-based collaborative filtering (UCF) performs the worst. Since this curation network is so sparse, users have provided insufficient implicit feedback for this recommender to figure out their similarities purely based on the curators they followed. Applying matrix factorization to find latent factors for both users and curators, BPR can outperform UCF by 11.9% on average for the top-k recall and precision, though it still lags the neural models. Relative to traditional collaborative filtering, we see that BPR's relaxed assumption of users' unobserved implicit feedback and pairwise ranking model is more effective for this task. Perhaps surprisingly, the Most Popular (MP) recommender performs slightly better than both UCF and BPR. Currently, Spotify suggests users with popular verified curators and friends on Facebook. Thus, although MP can result in better precision and recall than BPR and UCF, it necessarily ignores the long-tail challenge of uncovering curators who are not widely known.

NPR adds nonlinearity and improves over BPR by 20.3% without any auxiliary information. NPR also outperforms MP by 17.4% on average. The neural architectures are beneficial for this recommendation scenario. In addition, by aggregating the proposed curation style and coverage features, the final MCR model can further improve NPR by 5.7%. From the experiment, we observe the effectiveness of adopting a neural pairwise ranking model for the music curator ranking problem and incorporating curator *style* and *coverage*.

We compare with node2vec, which has demonstrated good performance in link prediction by efficiently exploring diverse neighborhoods on the graph. We find node2vec performs worse than MCR; indeed, it under-performs NPR by more than 10%. Both NPR and the node2vec method rely only on user-user (curators) following information, so the comparison can be fair. These results indicate that graph structure alone may be insufficient in this domain; in contrast, our MCR method can capture both graph structure (via the underlying matrix factorization which can uncover links between user-user and user-item pairs) and user preferences for curation style/coverage.

4.3 Comparison of Different Features

Here, we compare MCR with its simplified variants and the baseline NPR to examine how the curation style and coverage features perform in generating recommendations (see Table 1 *Normal Setting*). In the t-tests between NPR and MCR or its simplified variants on F1, we can get $p < 0.01$, which indicates a statistically significant difference.

Table 1. Comparison of different features of MCR under normal setting and cold start with **F1@K** scores.

Method	Normal setting				Cold start			
	@5	@10	@15	Ave Δ	@5	@10	@15	Ave Δ
NPR	.1082	.1003	.0889	–	.0615	.0651	.0585	–
A-MCR	.1117	.1028	.0907	+2.0%	.0744	.0753	.0626	+14.5%
C-MCR	.1125	.1034	.0913	+3.3%	.0818	.0696	.0709	+17.2%
S-MCR	.1127	.1037	.0913	+3.4%	.0781	.0753	.0692	+20.3%
MCR	.1124	.1043	.0920	+5.7%	.0818	.0753	.0676	+21.4%

Firstly, we observe that curation-based track embeddings (C-MCR) perform better than the audio-based features (A-MCR) in characterizing a curator's *coverage*. The track embeddings are learned from curation choices with skip-gram embedding model to extract the collective wisdom of curators, while the audio-based track features relying on audio analysis of individual tracks. The superior results from our track embeddings indicate the importance of curation decisions.

Additionally, the curation style features $\mathbf{e}^{style}$ (S-MCR) increase F1 score by 3.4% while the *coverage* with track embeddings (C-MCR) increasing it by 3.3%. Combining both the coverage and style, the proposed MCR framework is able to improve NPR by 5.7% in top-k F1 score. This nearly additive improvement suggests that coverage and style are complementary perspectives on curators and so both need to be properly modeled for curator recommendation.

4.4 Cold Start

In practice, it can be challenging to infer preferences of new users with very little feedback. We want to investigate whether the coverage and style contextual features can also help in the cold-start setting. For this experiment, we select users who follow fewer than 8 curators and examine how MCR and its simplified variants work for those users (in Table 1 *Cold-Start*). We find that the curating coverage with track embeddings (*C-MCR*) and the curation style (*S-MCR*) can improve NPR by 17.2% and 20.3% separately. Combining both of these features leads to an average improvement of 21.4% in F1 score. For each of the features and their combination, we see larger improvements compared with the original setting. This suggests that these features are critical in cold start scenarios, which are typical in real-world curation settings.

5 Conclusion

In this work, we tackle the problem of personalized music curator recommendation through a style-aware framework. We introduce the curation style and coverage features to capture a curator's individual approach for curating music. We also propose a curation-based embedding approach to capture inter-track agreement for music that pair well together. Through experiments, we observe that MCR results in the best precision, recall and F1 versus state-of-the-art. Also, the proposed style and coverage features can alleviate the challenges posed by cold start scenarios. In the future, we are interested in extending the models to support other curation platforms like Pinterest and Flipboard.

References

1. Ben-Elazar, S., et al.: Groove radio: a Bayesian hierarchical model for personalized playlist generation. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. ACM (2017)
2. Cao, D., Nie, L., He, X., Wei, X., Zhu, S., Chua, T.S.: Embedding factorization models for jointly recommending items and user generated lists. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 585–594. ACM (2017)
3. Celma, O.: Music recommendation. In: Music Recommendation and Discovery, pp. 43–85. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13287-2_3
4. Cheng, H.T., et al.: Wide & deep learning for recommender systems. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, pp. 7–10. ACM (2016)
5. Cheng, Z., Shen, J., Zhu, L., Kankanhalli, M., Nie, L.: Exploiting music play sequence for music recommendation. In: IJCAI: International Joint Conferences on Artificial Intelligence (2017)
6. Ding, D., Zhang, M., Li, S.Y., Tang, J., Chen, X., Zhou, Z.H.: BayDNN: friend recommendation with Bayesian personalized ranking deep neural network. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1479–1488. ACM (2017)
7. Fazel-Zarandi, M., Devlin, H.J., Huang, Y., Contractor, N.: Expert recommendation based on social drivers, social network analysis, and semantic data representation. In: Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems, pp. 41–48. ACM (2011)
8. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016)
9. Hannon, J., Bennett, M., Smyth, B.: Recommending Twitter users to follow using content and collaborative filtering approaches. In: Proceedings of the Fourth ACM Conference on Recommender Systems, pp. 199–206. ACM (2010)
10. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: WWW. International World Wide Web Conferences Steering Committee (2017)
11. Hinton, G.E.: Visualizing high-dimensional data using t-SNE. Vigiliae Christianae **9**(2), 2579–2605 (2008)

12. Jannach, D., Lerche, L., Kamehkhosh, I.: Beyond hitting the hits: generating coherent music playlist continuations with the right tracks. In: Proceedings of the 9th ACM Conference on Recommender Systems, pp. 187–194. ACM (2015)
13. Kitaya, K., Huang, H.H., Kawagoe, K.: Music curator recommendations using linked data. In: INTECH. IEEE (2012)
14. Liebman, E., Saar-Tsechansky, M., Stone, P.: DJ-MC: a reinforcement-learning agent for music playlist recommendation. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, pp. 591–599. International Foundation for Autonomous Agents and Multiagent Systems (2015)
15. Liu, Y., Chechik, D., Cho, J.: Power of human curation in recommendation system. In: Proceedings of the 25th International Conference Companion on World Wide Web, pp. 79–80. International World Wide Web Conferences Steering Committee (2016)
16. McDonald, D.W., Ackerman, M.S.: Expertise recommender: a flexible recommendation system and architecture. In: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, pp. 231–240. ACM (2000)
17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
18. Niu, W., Caverlee, J., Lu, H.: Neural personalized ranking for image recommendation. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. ACM (2018)
19. Van den Oord, A., Dieleman, S., Schrauwen, B.: Deep content-based music recommendation. In: Advances in Neural Information Processing Systems, pp. 2643–2651 (2013)
20. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. ELRA (2010)
21. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, pp. 452–461. AUAI Press (2009)
22. Sacheti, A., et al.: Recommending a content curator, US Patent App. 14/839,385, 2 March 2017
23. Schedl, M., Zamani, H., Chen, C.-W., Deldjoo, Y., Elahi, M.: Current challenges and visions in music recommender systems research. Int. J. Multimedia Inf. Retrieval **7**(2), 95–116 (2018). https://doi.org/10.1007/s13735-018-0154-2
24. Wang, D., Deng, S., Zhang, X., Xu, G.: Learning music embedding with metadata for context aware recommendation. In: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, pp. 249–253. ACM (2016)
25. Wang, X., Wang, Y.: Improving content-based and hybrid music recommendation using deep learning. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 627–636. ACM (2014)
26. Wu, Y., DuBois, C., Zheng, A.X., Ester, M.: Collaborative denoising auto-encoders for top-N recommender systems. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining. ACM (2016)
27. Zhao, Z., Cheng, Z., Hong, L., Chi, E.H.: Improving user topic interest profiles by behavior factorization. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1406–1416. International World Wide Web Conferences Steering Committee (2015)