Groove Radio: A Bayesian Hierarchical Model for Personalized Playlist Generation [GL: Current version is just over 9 pages without comments]

Anonymous Authors
Affiliation

ABSTRACT

This paper describes an algorithm designed for Microsoft's Groove music service, which serves millions of users world wide. We consider the problem of automatically generating personalized music playlists based on queries containing a "seed" artist and the listener's user ID. Playlist generation may be informed by a number of data sources including: listening patterns, genre taxonomy, acoustic features, and popularity. The importance assigned to each of these information sources may vary depending on the specific combination of user and "seed" artist.

The paper presents a method based on a variational Bayes solution for learning the parameters of a model containing a four-level hierarchy of global preferences, genres, sub-genres and artists. The proposed model further incorporates a personalization component for user-specific preferences. Empirical evaluations on both private and public datasets demonstrate the effectiveness of the algorithm and showcase the contribution of each of its components.[GL: Moved to intro: An online A/B experiment, comparing this algorithm to Groove's previous playlist algorithm, shows an increase in user listening time as well as a reduction in the number of skipped tracks.]

1. INTRODUCTION

Online music services such as Spotify, Pandora, Google Play Music and Microsoft's Groove serve as a major growth engine for today's music industry. A key experience is the ability to stream automatically generated playlists based on some criteria chosen by the listener. This paper considers the problem of automatically generating personalized music playlists based on queries containing a "seed" artist and the listener's user ID . We describe a solution designed for Microsoft's Groove internet radio service, which serves playlists to millions of users world wide.

In recommender systems research, collaborative filtering approaches such as matrix factorization are often used to learn relations between users and items [19]. The playlist generation task is essentially different as it requires learning a coherent sequence to allow smooth track transitions. Central to the approach taken in this research is the idea of estimating the relatedness of a "candidate" track to the previously selected tracks already in the playlist. Such relatedness can depend on multiple information sources, such as meta-data and domain semantics, the acoustic audio signal, and popularity patterns, as well as information extracted using collaborative filtering techniques.

The multiplicity of useful signals has motivated recent works [11, 25] to combine several information sources in order to model playlists. While it is known that all these factors play a role in the construction of a quality playlist, a key distinction of Groove's model is the idea that the importance of each of these information sources varies depending on the specific seed artist. For example, when composing a playlist for a Jazz artist such as John Coltrane, the importance of acoustic similarity features may be high. In contrast, in the case of a contemporary pop artist, such as Lady Gaga, features based on collaborative filtering may be more important. In Section 5, evaluations are provided in support of this assumption.

Another key element of the model in this paper is personalization based on the listener. For example, a particular user may prefer strictly popular tracks, while another prefers more esoteric content. Our method provides for modeling user specific preferences via personalization components.

The model in this paper is designed to support any artist in Groove's catalog, far beyond the small number of artists that dominate the lion's share of user listening patterns. The distribution of artists in the catalog contains a long tail of less popular artists for which insufficient information exists to learn artist specific parameters. Therefore, the Groove model also leverages the hierarchical taxonomy of genres, sub-genres and artists, native to the music domain. When particular artists or possibly even sub-genres are underrepresented in the data, the Groove model can still allow prediction by "borrowing" information from sibling nodes sharing the same parent in the taxonomy.

The Groove model casts playlist construction as a classification problem: predicting the next track given the seed artist, the user ID, and previously played tracks. As such, a hierarchical Bayesian model is suggested to perform playlist prediction using logistic regression. Parameter learning follows the variational Bayes approach of approximating the posterior distribution using a probability distribution which factorizes over the parameters. The evaluation of this model demonstrates the contribution of the hierarchical domain taxonomy and the personalization components to the predic-

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

tion task at hand. Beyond this evaluation, the performance of a production variant of approach described in this paper was validated in an online A/B experiment. The experiment shows an increase in user listening time and a reduction in the number of skipped tracks relative to the experiment baseline.

[GL: Added organization.. can be removed again if we are short on space] Our paper is organized as follows: Section 2 discusses relevant related work. Section 3 motivates our approach, discusses the specification of our model and explains the algorithm we apply to learn the parameters from data. Section 4 describes the features we use to encode playlist context. Section 5 gives the details of our experimental evaluation. Section 6 describes additional modifications that allow the proposed approach to work in large-scale scenarios. Finally, section 7 summarizes our results.

2. RELATED WORK

Automatic Playlist Generation is an active area of research and many formulations, approaches and evaluation frameworks have been proposed across the literature. In [6] the authors provide insight into how human playlists are constructed, suggesting that both audio similarity, domain similarity, and other semantic factors all play a role. throughout the literature the automatic playlist generation problem has been variously been formalized as: constraint satisfaction [29], the traveling salesman problem [18], and clustering in a metric space [28]. Several other works [14, 16, 34] opt for a recommendation oriented approach, ranking the best tracks in the catalog given the user and playlist context ("seed"). This is the approach we adopt in this paper. For a more in-depth discussion of the various approaches in this domain interested readers are referred to a survey of the literature by Bonnin and Jannach [4].

Several works have discussed computing similarity between musical tracks in various ways. Usage features, relating tracks that are consumed by overlapping sets of users users are featured prominently in several works [1, 23, 9]. Meta-Data features, relating tracks with similar semantic attributes, are also explored in many works [3, 24, 28, 32]. Audio features, relating tracks with similar audio signals has also been popular approach [8, 20, 34]. The prevalent approach in these works is to consider the mel frequency cepstrum coefficients (MFCC), and/or statistical properties thereof. Combining these various types of features to achieve improved similarity is also an active area of study [18, 23, 25, 31]. The methods of combining the features range from straightforward linear combinations [18] to learning a "similarity space" embedding [25].

Hierarchical classification has been previously explored in the literature. Several authors [5, 35] consider modified support vector machines which account for the the hierarchy of the domain taxonomy. Gopal et al. [13] propose a Bayesian hierarchical model multi-class classification model, similar to the one proposed in this work though notably lacking a personalization component, along with variational algorithms for learning the parameters.

The commercial music domain has a well-established hierarchical domain taxonomy, with layers (from coarse to fine) *genre*, *sub-genre*, *artist*, *album*, *track*. This taxonomy is in use for categorizing music in both brick-and-mortar and online music stores. The domain taxonomy has been used to enhance automated algorithms for genre classification [7], and music recommendations [9, 26].

Gillhofer and Schedl [12] empirically illustrate the importance of personalization, that is using information on the user, in selecting appropriate tracks. Personalization via latent space representation of users is a mainstay of classical recommendation systems [19]. Ferwerda and Schedl [11] propose modeling additional user features encoding personality and emotional state to enhance music recommendations.

In this work, we propose an approach that uses a Bayesian hierarchy to encode the domain taxonomy. Our approach allows a different combination of similarity features at each node in the hierarchy. Further, our approach models personal (i.e. per user) deviations from the hierarchical model. To the best of our knowledge, this paper is the first to propose an approach for automatic playlist generation which includes a hierarchical model that utilizes the domain taxonomy at various resolutions (genre, sub-genre and artist), considers multiple similarity types (audio, meta-data,usage), and incorporates personalization.

3. MODELING CONTEXTUAL PLAYLISTS

As was previously stated, the algorithm in this paper is designed to generate personalized playlists in the context of a "seed" artist and a specific listener. In Groove music, a playlist request named "radio" can be called by each subscriber using any artist in the catalog as a seed. An artist seed is a common scenario in many alternative online music services, but the algorithm in this paper can be extended to support also track seeds, genre seeds, seeds based on labels as well as various combinations of multi-seeds. In this paper we limit the discussion to the case of a single artist seed.

As explained earlier, constructing a playlist depends on multiple similarities used in estimating the relatedness of a new candidate track to the playlist. These similarities are based on meta-data, usage, acoustic features, and popularity patterns. However, the importance of each feature may be different for each seed. To account for difference in the importance of features for different seed artists, we propose a model which incorporates per-artist parameters that capture this variance. In the presence of a listener with historical listening data, the quality of the playlist may be further improved with personalization. Hence, the proposed model includes additional parameters that enable encoding per-user preferences.

A design goal of the proposed model is to support new or unknown listeners and/or seed artists which are new or "cold" (i.e. sparsely represented in the training data). If there is insufficient information on the listener, the proposed model performs a graceful fall-back, relying only on parameters related to the seed artist. In the case of an unknown or "cold" artist, the proposed model relies on the hierarchical domain taxonomy to allow a graceful fall-back using the parameters at the sub-genre level. This property applies also to underrepresented sub-genres and even genres, afforded by relying on correspondingly higher levels in the domain taxonomy.

The algorithm constructs a playlist by iteratively picking the next track to be added to the playlist. At each stage, the algorithm considers candidate tracks to be added to the playlist and predicts their relatedness to the seed, previously selected tracks and the listener. Previously selected tracks, together with seed artist and listener information, constitute the "context" from which the model is trained to predict the next track to play. This context is encoded as a feature vector, as described in Section 4.

3.1 Playlists as a Classification Problem

We denote by $\mathbf{x}_i \in \mathbb{R}^d$ the context feature vector representing the proposition of suggesting a particular track i at a particular "context" of previous playlist tracks, a seed artist and the listener.[GL: Added following ulrich's commnet..] Note that the features encoded in $\mathbf{x}_i$ will vary between different candidate tracks even if they belong to the same artist or genre. These context feature vectors are mapped to a binary label $r_i \in \{0, 1\}$,where $r_i = 1$ indicates a positive outcome for the proposition and $r_i = 0$ indicates a negative outcome. Thus, we have reduced our problem of playlist selection to a classification problem.

The dataset is denoted by $\mathcal{D}$ and consists of context vectors paired with labeled outcomes, denoted by the tuples $(\mathbf{x}_i, r_i) \in \mathcal{D}$. The binary labels may encode different real-world outcomes given the context. For example, in a dataset collected from playlist data logs, a track played to its conclusion may be considered a positive outcome ($r_i = 1$), whereas a track skipped mid-play may be considered a negative outcome ($r_i = 0$). Alternatively, consider a dataset of user-compiled collections. Tracks appearing in a collection may be considered positive outcomes, while some sample of catalog tracks not appearing in the collection are considered negative outcomes. In Section 5 we provide an evaluation using both of these approaches.

3.2 The Model

In what follows, we describe a hierarchical Bayesian classification model enriched with additional personalization parameters. We also provide a learning algorithm that generalizes the dataset $\mathcal{D}$ and enables generation of new playlists.

3.2.1 Notation

We discern vectors and matrices from scalars by denoting the former in **bold** letters. We further discern the vectors from the matrices by using lowercase letters for the vectors and capital letters for matrices. For example, x is a scalar, $\mathbf{x}$ is a vector and $\mathbf{X}$ is a matrix. We denote by $\mathbf{I}$ the identity matrix and $\mathbf{0}$ represents a vector of zeros.

The domain taxonomy is represented as a tree-structured graphical model. Each artist in the catalog corresponds to a leaf-node in the tree, with a single parent node corresponding to the artist's sub-genre. Similarly, each node corresponding to a sub-genre has a single parent node representing the appropriate genre. All nodes corresponding to genres have a single parent, the root node of the tree. We denote by $\texttt{par}(n)$ and $\texttt{child}(n)$ the mappings from a node indexed by n to its parent and to the set of its children, respectively. We denote by G, S, A and U the total number of genres, sub-genres, artists and users, respectively.

We denote by N the size of $\mathcal{D}$, our dataset. For the i'th tuple in $\mathcal{D}$, we denote by g_i, s_i, a_i and u_i the specific genre, sub-genre, artist and user (listener) corresponding to this datum, respectively. Finally, $\mathbf{w}^{(g)}_{g_i}$, $\mathbf{w}^{(s)}_{s_i}$, $\mathbf{w}^{(a)}_{a_i}$, $\mathbf{w}^{(u)}_{u_i}$, $\mathbf{w} \in \mathbb{R}^d$ denote the parameters for genre g_i, sub-genre s_i, artist a_i, user u_i, and the root, respectively.

3.2.2 Likelihood

We model the probability of a single example $(\mathbf{x}_i, r_i) \in \mathcal{D}$ given the context vector $\mathbf{x}_i$, the artist parameters $\mathbf{w}^{(a)}_{a_i}$ and the user parameters $\mathbf{w}^{(u)}_{u_i}$ as:

$$p(r_i|\mathbf{x}_i, \mathbf{w}^{(a)}_{a_i}, \mathbf{w}^{(u)}_{u_i}) = \left[\sigma\left(\mathbf{x}_i^\top(\mathbf{w}^{(a)}_{a_i} + \mathbf{w}^{(u)}_{u_i})\right)\right]^{r_i} \cdot \left[1 - \sigma\left(\mathbf{x}_i^\top(\mathbf{w}^{(a)}_{a_i} + \mathbf{w}^{(u)}_{u_i})\right)\right]^{(1-r_i)}, \quad (1)$$

where $\sigma(x) \overset{\text{def}}{=} (1+\mathrm{e}^{-x})^{-1}$ denotes the logistic function. The likelihood of the dataset $\mathcal{D}$ is simply the product of these probabilities i.e., $\prod_i p(r_i|\mathbf{x}_i, \mathbf{w}^{(a)}_{a_i}, \mathbf{w}^{(u)}_{u_i})$.

3.2.3 Hierarchical Prior

The prior distribution over the user parameters is a multivariate Normal: $p(\mathbf{w}^{(u)}_{u_i}|\tau_u) = \mathcal{N}(\mathbf{w}^{(u)}_{u_i}; \mathbf{0}, \tau_u^{-1}\mathbf{I})$, where τ_u is a precision parameter. The prior distribution over the root, genre, sub-genre, and artist parameters applies the domain taxonomy to define a hierarchy of priors as follows:

$$\begin{aligned}
p(\mathbf{w}^{(a)}_{a_i}|\mathbf{w}^{(s)}_{s_i}, \tau_a) &= \mathcal{N}(\mathbf{w}^{(a)}_{a_i}; \mathbf{w}^{(s)}_{s_i}, \tau_a^{-1}\mathbf{I}) \ , \\
p(\mathbf{w}^{(s)}_{s_i}|\mathbf{w}^{(g)}_{g_i}, \tau_s) &= \mathcal{N}(\mathbf{w}^{(s)}_{s_i}; \mathbf{w}^{(g)}_{g_i}, \tau_s^{-1}\mathbf{I}) \ , \\
p(\mathbf{w}^{(g)}_{g_i}|\mathbf{w}, \tau_g) &= \mathcal{N}(\mathbf{w}^{(g)}_{g_i}; \mathbf{w}, \tau_g^{-1}\mathbf{I}) \ , \\
p(\mathbf{w}|\tau_w) &= \mathcal{N}(\mathbf{w}; \mathbf{0}, \tau_w^{-1}\mathbf{I}) \ , \qquad (2)
\end{aligned}$$

where $\tau_a, \tau_s, \tau_g,\ \tau_w$ are scalar precision parameters. This prior structure allows sibling artists (belonging to the same sub-genre) to propagate information through the common sub-genre parameters. In case of a "cold" artist, it allows borrowing information from siblings by initializing the artist parameters according to the sub-genre parameters (i.e. "warm" initialization). As more information on the artist is observed, its parameters can gradually shift away from the prior position to a more accurate representation based on the observed data. These arguments follow along the hierarchical structure and apply to the sub-genre and genre parameters as well.

We define a set of hyper-priors over the precision parameters given by:

$$\begin{aligned}
p(\tau_u|\alpha, \beta) &= \mathcal{G}(\tau_u; \alpha, \beta) \ , & p(\tau_a|\alpha, \beta) &= \mathcal{G}(\tau_a; \alpha, \beta) \ , \\
p(\tau_s|\alpha, \beta) &= \mathcal{G}(\tau_s; \alpha, \beta) \ , & p(\tau_g|\alpha, \beta) &= \mathcal{G}(\tau_g; \alpha, \beta) \ , \\
p(\tau_w|\alpha, \beta) &= \mathcal{G}(\tau_w; \alpha, \beta) \ , & & \qquad (3)
\end{aligned}$$

where $\mathcal{G}(\tau; \alpha, \beta)$ is a Gamma distribution and α, β are global shape and rate hyper-parameters, respectively. We set $\alpha = \beta = 1$, resulting in a Gamma distribution with mean and variance equal to 1.

3.2.4 The Joint Probability

We collectively denote all the model's parameters by

$$\boldsymbol{\theta} \overset{\text{def}}{=} \Big\{\{\mathbf{w}^{(u)}_{k_u}\}^U_{k_u=1}, \{\mathbf{w}^{(a)}_{k_a}\}^A_{k_a=1}, \{\mathbf{w}^{(s)}_{k_s}\}^S_{k_s=1}, \{\mathbf{w}^{(g)}_{k_g}\}^G_{k_g=1}, \mathbf{w}, \tau_u, \tau_a, \tau_s, \tau_g, \tau_w\Big\},$$

and the hyper-parameters by $\mathcal{H} = \{\alpha, \beta\}$. The joint probability of the dataset $\mathcal{D}$ and the parameters $\boldsymbol{\theta}$ given the

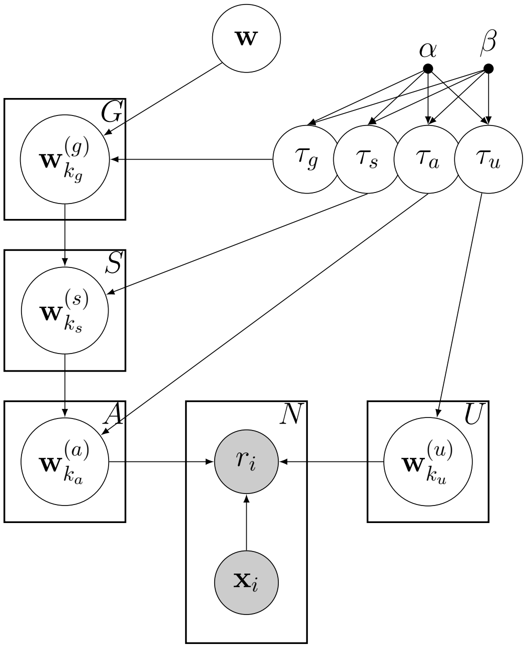


Figure 1: A graphical model representation of the proposed model. Unshaded circles denote unobserved random variables. Shaded circles denote observed random variables. Solid dots denote hyper-parameters.

hyper-parameters $\mathcal{H}$ is:

$$\begin{aligned} p\left(\mathcal{D}, \boldsymbol{\theta} \mid \mathcal{H}\right) = & \prod_{i=1}^{N} p\left(r_i \mid \mathbf{x}_i, \mathbf{w}_{a_i}^{(a)}, \mathbf{w}_{u_i}^{(u)}\right) \cdot \prod_{k_u=1}^{U} p(\mathbf{w}_{k_u}^{(u)} \mid \tau_u) \\ & \cdot \prod_{k_a=1}^{A} p(\mathbf{w}_{k_a}^{(a)} \mid \tau_a, \mathbf{w}_{\mathtt{par}(k_a)}^{(a)}) \cdot \prod_{k_s=1}^{S} p(\mathbf{w}_{k_s}^{(s)} \mid \tau_s, \mathbf{w}_{\mathtt{par}(k_s)}^{(g)}) \\ & \cdot \prod_{k_g=1}^{G} p(\mathbf{w}_{k_g}^{(g)} \mid \tau_g, \mathbf{w}) \cdot p(\mathbf{w} \mid \tau_w) \cdot \mathcal{G}\left(\tau_u; \alpha, \beta\right) \\ & \cdot \mathcal{G}\left(\tau_a; \alpha, \beta\right) \cdot \mathcal{G}\left(\tau_s; \alpha, \beta\right) \cdot \mathcal{G}\left(\tau_g; \alpha, \beta\right) \cdot \mathcal{G}\left(\tau_w; \alpha, \beta\right) . \end{aligned} \tag{4}$$

The graphical model representing this algorithm is depicted in Figure 1.

3.3 Variational Inference

We apply variational inference (or variational Bayes) to approximate the posterior distribution, $p(\boldsymbol{\theta}|\mathcal{D},\mathcal{H})$, with some distribution $q(\boldsymbol{\theta})$, by maximizing the (negative) variational free energy given by $\mathcal{F}[q(\boldsymbol{\theta})] \overset{\text{def}}{=} \int q(\boldsymbol{\theta}) \log \frac{p(\mathcal{D},\boldsymbol{\theta}|\mathcal{H})}{q(\boldsymbol{\theta})} \, \mathrm{d}\boldsymbol{\theta}$. $\mathcal{F}$ serves as a lower bound on the log marginal likelihood, or logarithm of the model evidence.

3.3.1 Logistic Bound

The joint probability in (4) includes Gaussian priors which are not conjugate to the likelihood, due to the sigmoid functions appearing in (1). In order to facilitate approximate inference, these sigmoid functions are bounded by a "squared exponential" form, which is conjugate to the Gaussian prior. The following derivations resemble variational inference for logistic regression as described in more detail in [15].

First, the sigmoid functions appearing in (4) are lower-bounded using the logistic bound [17]. Introducing an additional variational parameter ξ_i on each observation i allows the following bound:

$$\sigma(h_i)^{r_i} \cdot [1 - \sigma(h_i)]^{1-r_i} \geq e^{r_i h_i} \left[\sigma(\xi_i) e^{-\frac{1}{2}(h_i + \xi_i) - \lambda_i (h_i^2 + \xi_i^2)}\right] \tag{5}$$

where $h_i \overset{\text{def}}{=} \mathbf{x}_i^\top(\mathbf{w}_{a_i}^{(a)} + \mathbf{w}_{u_i}^{(u)})$ and $\lambda_i \overset{\text{def}}{=} \frac{1}{2\xi_i}[\sigma(\xi_i) - \frac{1}{2}]$. Using (5), we substitute for the sigmoid functions in $p(\mathcal{D},\boldsymbol{\theta}|\mathcal{H})$ to obtain the lower bound $p_\xi(\mathcal{D},\boldsymbol{\theta}|\mathcal{H})$. We can apply this bound to the variational free energy:

$$\mathcal{F}[q(\boldsymbol{\theta})] \geq \mathcal{F}_\xi[q(\boldsymbol{\theta})] \overset{\text{def}}{=} \int q(\boldsymbol{\theta}) \log \frac{p_\xi(\mathcal{D},\boldsymbol{\theta}|\mathcal{H})}{q(\boldsymbol{\theta})} \, \mathrm{d}\boldsymbol{\theta} \; .$$

The analytically tractable $\mathcal{F}_\xi[q(\boldsymbol{\theta})]$ is used as our optimization objective with respect to our approximate posterior distribution $q(\boldsymbol{\theta})$.

3.3.2 Update Equations

Variational inference is achieved by restricting the approximation distribution $q(\boldsymbol{\theta})$ to the family of distributions that factor over the parameters in $\boldsymbol{\theta}$. With a slight notation overloading for q we have

$$\begin{aligned} q(\boldsymbol{\theta}) = & \prod_{k_u=1}^{U} q(\mathbf{w}_{k_u}^{(u)}) \cdot \prod_{k_a=1}^{A} q(\mathbf{w}_{k_a}^{(a)}) \cdot \prod_{k_s=1}^{S} q(\mathbf{w}_{k_s}^{(s)}) \cdot \prod_{k_g=1}^{G} q(\mathbf{w}_{k_g}^{(g)}) \\ & \cdot q(\mathbf{w}) \cdot q(\tau_u) \cdot q(\tau_a) \cdot q(\tau_s) \cdot q(\tau_g) \cdot q(\tau_w), \end{aligned} \tag{6}$$

where q denotes a different distribution function for each parameter in $\boldsymbol{\theta}$.

Optimization of $\mathcal{F}_\xi$ follows using coordinate ascent in the function space of the variational distributions. Namely, we compute functional derivatives $\partial\mathcal{F}_\xi/\partial q$ with respect to each distribution q in (6). Equating the derivatives to zero, together with a Lagrange multiplier constraint to make q integrate to 1 (a distribution function), we get the update equations for each q in (6). At each iteration, the optimization process alternates through parameters, applying each update equation in turn. Each such update increases the objective $\mathcal{F}_\xi$, thus increasing $\mathcal{F}$. We continue to iterate until convergence. Owing to the analytical form of $\mathcal{F}_\xi$ and the factorization assumption on the approximation distribution q, each component of q turns out to be Gaussian distributed, in the case of the weight parameters, or Gamma distributed, in the case of the precision parameters. Thus, in the following we describe the update step of each component of q in terms of its canonical parameters.

Update for user parameters

For each user $k_u = 1 \dots U$ we approximate the posterior of $\mathbf{w}_{k_u}^{(u)}$ with a Normal distribution

$$q(\mathbf{w}_{k_u}^{(u)}) = \mathcal{N}\left(\mathbf{w}_{k_u}^{(u)}; \boldsymbol{\mu}_{k_u}^{(u)}, \boldsymbol{\Sigma}_{k_u}^{(u)}\right), \tag{7}$$

$$\begin{aligned} \boldsymbol{\Sigma}_{k_u}^{(u)} &= \left[\tau_u \mathbf{I} + \sum_{i=1}^{N} \mathbb{I}\left[u_i = k_u\right] 2\lambda_i \cdot \mathbf{x}_i \mathbf{x}_i^\top\right]^{-1} \\ \boldsymbol{\mu}_{k_u}^{(u)} &= \boldsymbol{\Sigma}_{k_u}^{(u)} \cdot \left[\sum_{i=1}^{N} \mathbb{I}\left[u_i = k_u\right] \left(r_i - \frac{1}{2} - 2\lambda_i \mathbf{x}_i^\top \langle \mathbf{w}_{a_i}^{(a)} \rangle\right) \mathbf{x}_i\right], \end{aligned}$$

where $\mathbb{I}\left[\cdot\right]$ is an indicator function. The angular brackets are used to denote an expectation over $q(\boldsymbol{\theta})$ i.e., $\langle \mathbf{w}_{a_i}^{(a)} \rangle = \mathbb{E}_{q(\boldsymbol{\theta})}[\mathbf{w}_{a_i}^{(a)}]$

Update for artist parameters

For each artist $k_a = 1 \dots A$ we approximate the posterior of $\mathbf{w}^{(a)}_{k_a}$ with a Normal distribution

$$q(\mathbf{w}^{(a)}_{k_a}) = \mathcal{N}\left(\mathbf{w}^{(a)}_{k_a}; \boldsymbol{\mu}^{(a)}_{k_a}, \boldsymbol{\Sigma}^{(a)}_{k_a}\right), \tag{8}$$

$$\begin{aligned}
\boldsymbol{\Sigma}^{(a)}_{k_a} &= \left[\tau_a \cdot \mathbf{I} + \sum_{i=1}^{N} \mathbb{I}\left[a_i = k_a\right] 2\lambda_i \cdot \mathbf{x}_i \mathbf{x}_i^{\top}\right]^{-1}, \\
\boldsymbol{\mu}^{(a)}_{k_a} &= \boldsymbol{\Sigma}^{(a)}_{k_a} \cdot \Bigg[\tau_a \langle \mathbf{w}^{(s)}_{\texttt{par}(k_a)} \rangle + \\
&\qquad \sum_{i=1}^{N} \mathbb{I}\left[a_i = k_a\right] \left(r_i - \frac{1}{2} - 2\lambda_i \mathbf{x}_i^{\top} \langle \mathbf{w}^{(u)}_{u_i} \rangle\right) \mathbf{x}_i\Bigg].
\end{aligned}$$

Update for sub-genre parameters

For each sub-genre $k_s = 1 \dots S$ we approximate the posterior of $\mathbf{w}^{(s)}_{k_s}$ with a Normal distribution

$$q(\mathbf{w}^{(s)}_{k_s}) = \mathcal{N}\left(\mathbf{w}^{(s)}_{k_s}; \boldsymbol{\mu}^{(s)}_{k_s}, \boldsymbol{\Sigma}^{(s)}_{k_s}\right), \tag{9}$$

$$\begin{aligned}
\boldsymbol{\Sigma}^{(s)}_{k_s} &= (\tau_s + |\mathcal{C}_{k_s}| \tau_a)^{-1} \cdot \mathbf{I}, \\
\boldsymbol{\mu}^{(s)}_{k_s} &= \boldsymbol{\Sigma}^{(s)}_{k_s} \cdot \left[\tau_s \langle \mathbf{w}^{(g)}_{\texttt{par}(k_s)} \rangle + \tau_a \sum_{k_a \in \mathcal{C}_{k_s}} \langle \mathbf{w}^{(a)}_{k_a} \rangle\right],
\end{aligned}$$

where $\mathcal{C}_{k_s} = \texttt{child}(k_s)$ is the set of artists in sub-genre k_s.

Update for genre parameters

For each genre $k_g = 1 \dots G$ we approximate the posterior of $\mathbf{w}^{(g)}_{k_g}$ with a Normal distribution

$$q(\mathbf{w}^{(g)}_{k_g}) = \mathcal{N}\left(\mathbf{w}^{(g)}_{k_g}; \boldsymbol{\mu}^{(g)}_{k_g}, \boldsymbol{\Sigma}^{(g)}_{k_g}\right), \tag{10}$$

$$\begin{aligned}
\boldsymbol{\Sigma}^{(g)}_{k_g} &= \left(\tau_g + |\mathcal{C}_{k_g}| \tau_s\right)^{-1} \cdot \mathbf{I}, \\
\boldsymbol{\mu}^{(g)}_{k_g} &= \boldsymbol{\Sigma}^{(g)}_{k} \cdot \left[\tau_g \langle \mathbf{w} \rangle + \tau_s \sum_{k_s \in \mathcal{C}_{k_g}} \langle \mathbf{w}^{(s)}_{k_s} \rangle\right],
\end{aligned}$$

where $\mathcal{C}_{k_g} = \texttt{child}(k_g)$ is the set of sub-genres for genre k_g.

Update for the root parameters

We approximate the posterior over $\mathbf{w}$ with a Normal distribution

$$q(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right), \tag{11}$$

$$\boldsymbol{\Sigma} = (\tau_w + \tau_g \cdot G)^{-1} \cdot \mathbf{I} \quad \text{and} \quad \boldsymbol{\mu} = \Sigma \cdot \left[\tau_g \sum_{k_g=1}^{G} \langle \mathbf{w}^{(g)}_{k_g} \rangle\right].$$

Update for the precision parameters

The model includes 5 precision parameters: $\tau_u, \tau_a, \tau_s, \tau_g, \tau_w$. Each is approximated with a Gamma distribution. For the sake of brevity we will only provide the update for τ_u. We approximate its posterior with $q(\tau_u) = \mathcal{G}(\tau_u \mid \alpha_u, \beta_u)$, where $\alpha_u = \alpha + \frac{d \cdot U}{2}$ is the shape and $\beta_u = \beta + \frac{1}{2} \sum_{k_u=1}^{U} \langle \left(\mathbf{w}^{(u)}_{k_u}\right)^{\top} \mathbf{w}^{(u)}_{k_u} \rangle$ is the rate.

Update for variational parameters

The variational parameters $\xi_1 \dots \xi_N$ in $\mathcal{F}_\xi$ are chosen to maximize $\mathcal{F}_\xi$ such that the bound on $\mathcal{F}$ is tight. This is achieved by setting $\xi_i = \sqrt{\langle \left(\mathbf{x}_i^{\top}\left(\mathbf{w}^{(a)}_{a_i} + \mathbf{w}^{(u)}_{u_i}\right)\right)^2 \rangle}$. We refer the reader to Bishop [2] for a deeper discussion.

3.4 Prediction and Ranking

At run time, given a "seed" artist a^* and a user u^*, our model computes the probability of a positive outcome for each context vector $\mathbf{x}_1 \dots \mathbf{x}_M$ corresponding to M possible tracks. This probability is given by:

$$\begin{aligned}
\hat{r}_m &\overset{\text{def}}{=} p(r_m = 1 \mid \mathbf{x}_m, \mathcal{D}, \mathcal{H}) \\
&\approx \int \sigma(h_m) q(\boldsymbol{\theta}) d\boldsymbol{\theta} = \int \sigma(h_m) \mathcal{N}\left(h_m \mid \mu_m, \sigma_m^2\right) dh_m
\end{aligned} \tag{12}$$

where the random variable h_m has a Normal distribution:

$$\begin{aligned}
&h_m = \mathbf{x}_m^{\top}\left(\mathbf{w}^{(u)}_{u^*} + \mathbf{w}^{(a)}_{a^*}\right) \sim \mathcal{N}\left(h_m \mid \mu_m, \sigma_m^2\right), \\
&\mu_m \overset{\text{def}}{=} \langle \mathbf{x}_m^{\top}\left(\mathbf{w}^{(u)}_{u^*} + \mathbf{w}^{(a)}_{a^*}\right) \rangle, \quad \sigma_m^2 \overset{\text{def}}{=} \langle \left(\mathbf{x}_m^{\top}\left(\mathbf{w}^{(u)}_{u^*} + \mathbf{w}^{(a)}_{a^*}\right) - \mu_m\right)^2 \rangle.
\end{aligned}$$

Finally, the integral in (12) is approximated following MacKay [22] using

$$\int \sigma(h_m) \mathcal{N}\left(h_m \mid \mu_m, \sigma_m^2\right) dh_m \approx \sigma\left(\mu_m / \sqrt{1 + \pi \sigma_m^2 / 8}\right). \tag{13}$$

4. FEATURES FOR ENCODING CONTEXT

The model as described in the previous section makes no assumptions on the nature of the contextual features beyond the fact that they encode relevant information for choosing the next track to be added to the playlist. Since the main contribution of this paper is in the definition of the model and corresponding learning algorithm, our efforts to find the best features for the application of playlist generation are by no means exhaustive. However, in this section we offer some insights into the types of features used by the algorithm.

In general, the features encode different types of similarities or relations comparing a candidate track to be added to the playlist with tracks already selected as well as with the seed artist. In cases where specific similarities are not applicable we apply zero-imputation. [GL: Added following ulrich's comment:] In our practice the number of features, denoted d below, is on the order of 10-20. We divide these features into four groups categorized according to the type of signal employed in their calculation: [GL: Added more formal description of features]

Acoustic Audio Features (AAF) - Let $\mathcal{D}_a = \left\{\mathbf{x_i} \in \mathbb{R}^d\right\}_{i=1}^{N_a}$ denote the audio samples (encoded as a vector of continuous features) for a particular artist a. Let $\hat{\theta}_a = \arg\max_\theta \prod_i p\left(x_i \mid \theta\right)$ denote the maximum likelihood parameter setting for the (Gaussian Mixture Model) GMM for the distribution of acoustic audio features. The audio similarity between two artists a_1 and a_2 is then given by $\mathrm{KL}\left[p\left(\mathbf{x} \mid \theta_{a_1}\right) \mid\mid p\left(\mathbf{x} \mid \theta_{a_2}\right)\right]$), the Kullback Liebler divergence between the two corresponding distributions. To compute the features $\mathbf{x}_i$, we compute the mel frequency cepstrum coefficients (MFCC). This approach follows [30] and can be applied similarly to compute the similarity between audio tracks.

Usage Features (UF) - Following [27] we learn a low-rank factorization of the matrix $\mathcal{R}$, where $r_{i,j}$ the element on the i-th row and j-th column denotes the binary rating given to the j-th artist in the catalog by the i-th user of the system. This formulation is parameterized by a k-dimensional vector for each user and artist appearing in the training data. More precisely, the user vectors $\{\mathbf{u}_i\}_{i=1}^{|U|}$ and artist vectors $\{\mathbf{a}_j\}_{j=1}^{|A|}$, collectively denoted θ_u are chosen to optimize the Bayesian model $p(\mathcal{D}_u, \theta_u)$ (Equation (5) in [27]) over the training dataset containing binary interactions between users and artists, denoted $\mathcal{D}_u$. The similarity between two artists a_1 and a_2 is then given by $\sigma\left(\mathbf{a}_{a_1}^{\top}\mathbf{a}_{a_2}\right)$, where σ denotes the sigmoid function. A similar treatment can be applied to a matrix of user-track interaction and used to derive similarities between tracks.

Meta-Data Features (MDF) - Meta-Data features are based on the semantic tags associated with different tracks and artists in the catalog(e.g. "easy-listening", "upbeat", "90's"). We obtain this information from a third-party dataset. Each artist j in the catalog is encoded as a vector $\mathbf{b}_j \in \{0,1\}^{|V|}$, where V denotes the set of possible binary semantic tags. These vectors are generally sparse, as each artist corresponds to only a small number of semantic tags. The similarity between two artists a_1 and a_2 is then given by the cosine similarity:

$$\frac{\mathbf{b}_{a_1}^{\top}\mathbf{b}_{a_2}}{\parallel \mathbf{b}_{a_1} \parallel \parallel \mathbf{b}_{a_2} \parallel}$$

Track to track similarity can be computed by using sparse vectors representing tracks.

Popularity Features (PF) - Popularity is a measure of the prevalence of a track or artist in the dataset. Popularity is used to compute unary features representing the popularity of a candidate track and its artist. For a particular artist a_1 such a feature is computed $\text{POP}_{a1} = \frac{\#n_{\mathcal{D}_u}(a_1)}{|\mathcal{D}_u|}$, where $\#n_{\mathcal{D}_u}(a_1)$ denotes the number of users who consumed a track by artist a_1 in the training corpus $\mathcal{D}_u$. Pairwise popularity features are computed relating the popularity of the candidate track and its artist to the popularity of the seed artist and previous tracks. The relative popularity of artists a_1 and a_2 can be computed as $\frac{\text{POP}_{a2}}{\text{POP}_{a1}}$. track to track and track to artist relative popularity can be computed similarly.

5. EVALUATION

The following section describes the evaluation procedure applied to the proposed algorithm.

5.1 Datasets

As explained in Section 3.1, the model in this paper treats playlist generation as a classification problem, for which the parameters can be learned from examples, judiciously labeled through a variety of approaches. The two datasets used for the evaluations exemplify different approaches to the construction of the training data:

Groove Music Dataset is a proprietary dataset of user preferences that was collected anonymously from Microsoft's Groove music service. It contains 334,120 users, 472,908 tracks from 45,239 artists categorized into 100 sub-genres from 17 genres. Positive labels are assigned to tracks in a user's listening history that were heard to completion. Negative labels are assigned to tracks that were skipped in mid-play by the user.

30Music Dataset is a publicly available dataset of user playlists [33]. Tracks in this dataset were intersected with Groove's catalog in order to enable feature generation (i.e. audio and content features). The resulting dataset contains 14,185 users, 252,424 tracks from 63,314 artists categorized into 99 sub-genres from 17 genres. Positive labels are assigned to tracks appearing in a user's playlist. Since no skip information was available, negatively labeled examples were obtained by uniformly sampling from tracks that did not appear in the user's playlist.

5.2 Experimental Setup and Results

We quantify the quality of the model using the *Area Under Curve* (AUC) metric [10]. AUC enables quantifying the trade-off between true positives and false positives over a range of threshold configurations. By doing so, AUC captures the overall quality of a particular prediction method.

The model was trained on 70% of the examples and evaluated on the remaining 30%. [GL: We may have more results using cross-validation later in the week, so we can claim a stronger experimental setup] We compare it against several baselines:

Partial Hierarchy - This baseline (actually three separate baselines, one for each level of the hierarchy) learns only a subset of the domain taxonomy parameters, corresponding to increasingly coarser levels in the hierarchy than the four levels of the the full model describe in Section 3. Evaluations are provided at each of the sub-hierarchies: global (no hierarchy), genre (two-level) , and sub-genre (three-level).

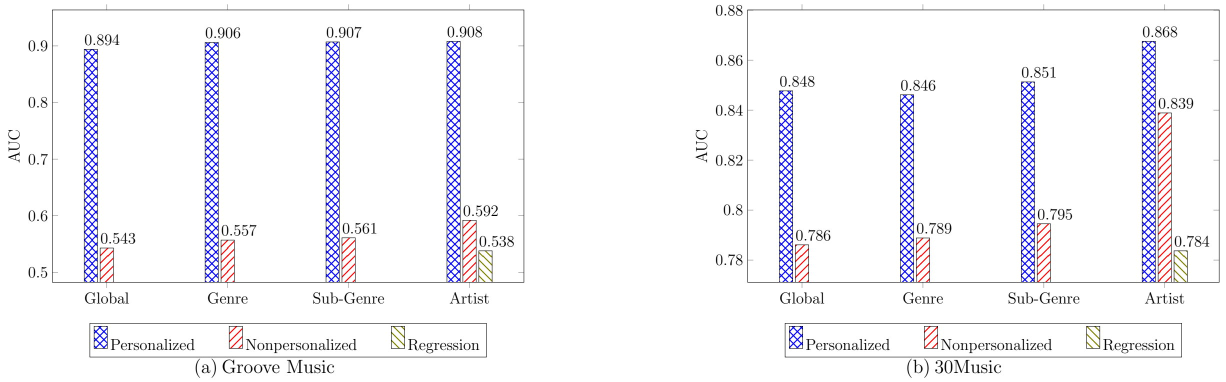
Non-Personalized - A non-personalized version of the Groove model is evaluated by removing the per-user parameters. [GL: Added combined model part.. relating different parametrizations to works from the literature]

Combined Models The framework proposed in this work spans a family of models parameterized by $\eta \in \{1,2,3,4\}$, the number of layers of the hierarchical taxonomy and $\varphi \in \{0,1\}$, the absence/presence of a personalization component. Viewed this way, we can roughly equate different configurations of these hyper-params with previous approaches proposed in the literature. For instance the full hierarchy, non-personalized ($\eta = 4, \varphi = 0$) variant of the model roughly corresponds to the approach proposed in [13]. Further, the personalized no-hierarchy variant ($\eta = 1, \varphi = 1$), corresponds to methods that have attempted to combine different similarity signals without taking into account user preferences [18, 25]. Finally, a two-layer hierarchy with personalization ($\eta = 2, \varphi = 1$) is similar in spirit (though not identical in formulation) to methods that explicitly model biases for different genres [9, 26].

Non-Personalized Regression - This baseline is a simplified version of the model where the likelihood is changed to consider a regression problem and the optimization is based on an maximum a posteriori (MAP) solution (instead of variational Bayes). This method also considers the domain taxonomy, but results are provided only at the artist level.

Figure 2 depicts the results of the personalized variant of the model vs. the non-personalized baseline, considering gradually increasing partial domain taxonomies along the x-axis of the figure. *Global*, *Genre*, and *Sub-Genre* labels,denotes a one, two , and three level taxonomy, respectively. The *Artist* label denotes the full-hierarchy model shown in Figure 1.

The effect of adding personalization is apparent in both



(a) Groove Music

(b) 30Music

Figure 2: The effects of personalization and hierarchy on accuracy (as measured by AUC) are illustrated for (a) the Groove Music dataset (a) the 30Music Dataset. We evaluate the personalized and non-personalized approach at each level of the partial hierarchy (increasing depth from left to right). The regression approach is defined only for the full hierarchy (see text).

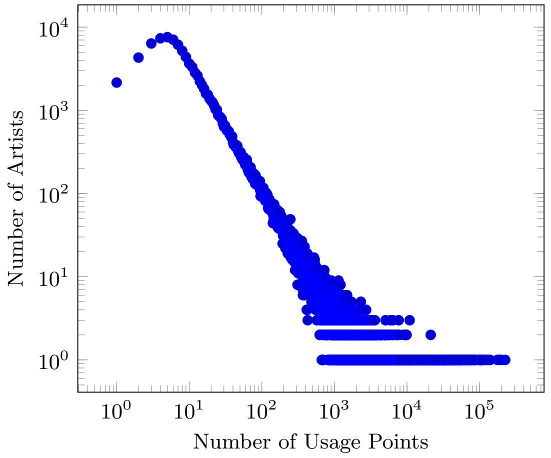


Figure 3: A log-log histogram of usage points for artists in the groove catalog (computed over a sub-set). The large majority of artists have very few observed usage points.

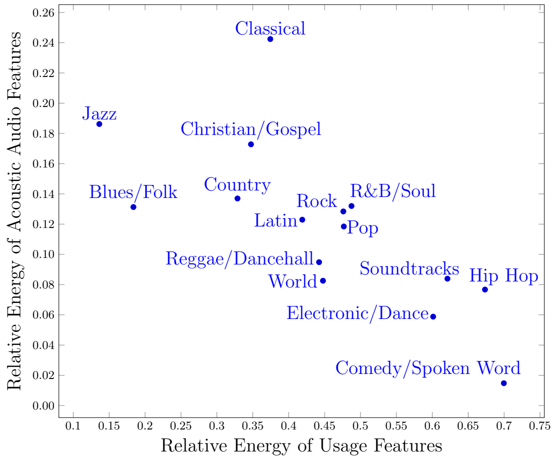


Figure 4: The relative energy of the weights assigned to each Usage and Audio Features for the different musical genres considered.

datasets, but more notable in the Groove Music dataset where "skips", as opposed to random sampling, are used to define negative labels. This indicates that skipping of tracks has more user-specific dependencies beyond the user's preference for a particular artist or genre. Also, clearly visible in both datasets is the importance of the hierarchical taxonomy: AUC results improve as additional levels of the hierarchy are considered. This effect is more considerable in the 30Music dataset, where we conjecture that the prediction task is based on a "less personal" signal. The non-personalized regression baseline performs worse than the proposed model for both datasets. We conjecture this is due to the fact that this model is trained using MAP and does not allow the consideration of the inherent uncertainty at the different levels of the hierarchy afforded by the Bayesian model.

We also considered the contribution of the different feature groups defined in Section 4. To this end we trained the model, using the Groove Music dataset, on each subset of the features separately and evaluated the AUC. Note that for this analysis we used a subset of the data for which no features had missing values.

Table 1 summarizes the results of this analysis. Using only usage features results in the highest AUC score, followed by popularity, meta-data and acoustic audio features, respectively. Although usage features perform well, on their own, we get additional benefits from using other types of features. This is especially relevant when we consider "cold" artists that have little usage information available. These artists are by far the majority of those appearing in the catalog as can be seen in the histogram (computed over a subset of usage data) shown in Figure 3. Furthermore, the importance of usage features in relation to other features varies across the domain taxonomy. Figure 4 illustrates this variance. The figure shows that geners such as *Classical* and *Jazz* rely (relatively) much more on audio features and less on usage features than genres such as *Spoken Word* and *Hip Hop*.

	AAF	**MDF**	**PF**	**UF**	**All**
AUC	0.843	0.855	0.865	0.871	0.874

Table 1: AUC achieved by training only on a subset of features on the Groove Music prediction dataset. Feature groups are defined in Section 4

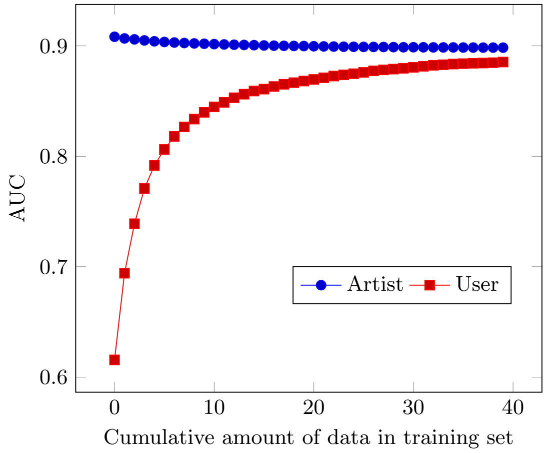


Figure 5: A cumulative plot illustrating the effect of Artist/User coldness on test AUC. The AUC is plotted for all Artists/Users with at least x labeled data points in the training set.

In recommendation systems the "cold" user/item problem describes the difficulty of providing recommendations for users/items with little to no previous interaction with the system. Figure 5 plots the AUC on the Groove Music dataset as a function of the amount of data available for users and artists, respectively. The plot is cumulative e.g., at value 10 along the X-axis all users / artists with at most 10 training examples are considered. AUC levels are significantly lower for cold users but quickly improve as the number of training examples increases. This trend is another indication of the importance of personalized information. In contrast to users, there is almost no change in artists' AUC values per different number of observations, and even artists with zero training examples show high AUC values. This is enabled due to the models use of the domain taxonomy to mitigate the "cold" artist problem.

Finally, Figure 6 provides an illustration of artist parameters using a t-SNE embedding [21] of artist parameter mean values from the Groove music dataset. Note that proximity in this plot is determined by similarity in the parameters of the learning model. Namely, it does not necessarily reflect "musical similarity", but instead it indicates similarity in the importance of the contextual features. The fact that many artists of the same genre do not cluster together supports the model's underlying assumption from Section 1 that different considerations needs to be applied when generating their playlists. It also suggests that priors at the genre level alone are too coarse and must be broken down to their subgenres.

5.3 Anecdotal Results

To give a flavor of the type of output generated by the proposed approach Table 2 shows the top five tracks in the playlist for three very different seed artists. Notably, using the pop star *Rihanna* as a seed results in a playlist composed of tracks by other pop-music artist which do not necessarily sound similar to *Rihanna*. For the Jazz and Classical seed artists, we see that the playlist is not only composed of tracks from the same genre of the artist, but further many tracks include instrumentation similar to that of the seed artist. These playlists are generated by a varaint of the algorithm

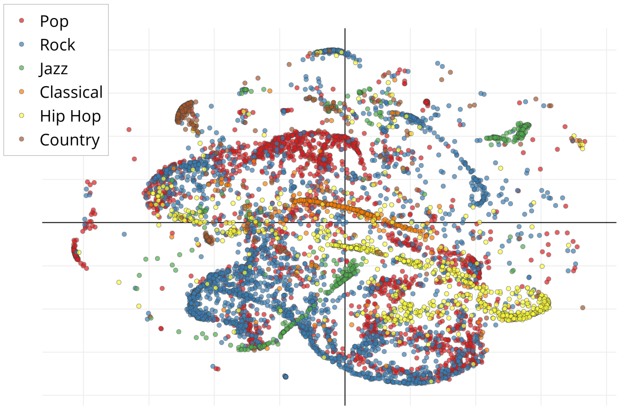


Figure 6: tSNE embedding of artist parameters. The figure shows a complex manifold that could not be captured by modeling higher levels of the taxonomy alone.

Track Title	Artist Name	Album Name
Artist Seed: Rihanna (Pop Singer)		
Yeah, I Said It	Rihanna	ANTI
Birthday	Katy Perry	PRISM
She Will Be Loved	Maroon 5	Songs About Jane
I'm Real	Jennifer Lopez	J. Lo
Yeah! (feat. Lil' Jon & Ludacris)	Usher	Confessions
Artist Seed: Wes Montogomery (Jazz Guitarist)		
Bumpin' On Sunset	Wes Montgomery	Wes Montgomery: Finest Hour
A Go Go	John Scofield	A Go Go
The Natives Are Restless Tonight	Horace Silver	Song For My Father
I Remember Clifford	Lee Morgan	The Ultimate Collection
Gee Baby, Ain't I Good To You	Kenny Burrell	Midnight Blue
Artist Seed: Itzhak Perlman (Classical Violinist)		
Il Postino: Theme (Instrumental)	Itzhak Perlman	Cinema Serenade
Brahms: Hungarian Dance No.5	Nicola Benedetti	The Violin
Violin Concerto No. 2 in E Major ..	Joshua Bell	Bach
Piezas Caracteristicas, Op. 92	John Williams	The Ultimate Guitar Collection
Adagio for Strings	Leonard Bernstein	Barber's Adagio ...

Table 2: Anecdotal results: shows the top 5 tracks in the playlist generated for several seed seed artists.

proposed in this work which currently powers the Microsoft's *Groove Music* service.

[GL: Removed section on AB Testing...]

6. PRACTICAL CONSIDERATIONS

In this section we offer some discussion on ideas that allow the application of the model proposed in this paper to a real-world system serving a large number of users.

The playlist is constructed sequentially by picking the next track using the ranking induced by $\hat{r}_m$ from (13). However, in practice we first apply two important heuristics. First, since it is impractical to consider the tens of millions of tracks in the Groove music catalog, we first pre-compute a candidate list of $M \approx 1,000$ tracks for each possible "seed" artist. The candidate list for an artist a^* consists of a^*'s tracks and tracks from artists similar to a^*. Second, we define a Boltzmann distribution over the $m = 1 \ldots M$ tracks with each candidate track having a probability given by $p_m = \frac{e^{s\cdot\hat{r}_m}}{\sum_{i=1}^{M} e^{s\cdot\hat{r}_i}}$, where s is a tunable parameter. The next track is chosen randomly with probability p_m. This scheme ensures a degree of diversity, controlled by s, between multiple instances of similar playlists. This type of randomization also acts as an exploration mechanism, allowing labels to be collected from areas where the model is less certain. This reduces the feedback loop effect when learning future models based on user interactions with the system.

An advantage of the Bayesian setup described in this work is that it is fairly straightforward to adjust the model parameters in an online fashion. For example, consider the scenario where a playlist user skips several tracks in a row. Our approach could be extended to update the user parameter vector given these additional explicit negatives, before computing the next track selection.

Finally, our model is designed for implicit feedback signals, as these are more common in commercial settings, hence the use of binary labels. However, in some scenarios explicit user ratings are known. Support for such scenarios can be achieved by modifying the likelihood term of our model (in (1)) and re-deriving the update equations.

7. CONCLUSION

This work describes a model for playlist generation designed for Microsoft's Groove music service. The model incorporates per-artist parameters in order to capture the unique characteristics of an artist's playlists. The domain taxonomy of genres, sub-genres and artists is utilized in order to allow training examples from one artist to inform predictions of other related artists. Hence, the model offers good predictions even when a playlist is requested for a "seed" artist for whom little (or even no) historical information is available. Furthermore, the proposed model is endowed with the capacity to capture particular user preferences for those users who are frequent playlist requesters, enabling a personalized playlist experience.

A variational inference learning algorithm is applied and a detailed description of the update steps of each parameter is provided in Section 3. The evaluations in Section 5 show the benefit of using the domain taxonomy as well as the personalization components for the playlist generation task at hand. Finally, an online experiment is described showing a significant increase in user listening time and a significant reduction in skipped tracks.

8. REFERENCES

[1] O. Barkan and N. Koenigstein. Item2vec: Neural item embedding for collaborative filtering. *arXiv preprint arXiv:1603.04259*, 2016.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[3] 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.*, 2013.

[4] G. Bonnin and D. Jannach. Automated generation of music playlists: Survey and experiments. *ACM Comput. Surv.*, 2015.

[5] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines categories and subject descriptors. In *Proceedings of CIKM Conference*, 2004.

[6] S. J. Cunningham, D. Bainbridge, and A. Falconer. More of an Art than a Science': Supporting the Creation of Playlists and Mixes. In *Proc. ISMIR*, 2006.

[7] C. Decoro, Z. Barutcuoglu, and R. Fiebrink. Bayesian aggregation for hierarchical genre classification. In *Proceedings of ISMIR Conference*, 2007.

[8] M. Dopler, M. Schedl, T. Pohle, and P. Knees. Accessing music collections via representative cluster prototypes in a hierarchical organization scheme. In *Proceedings of ISMIR Conference*, 2008.

[9] G. Dror, N. Koenigstein, and Y. Koren. Yahoo ! music recommendations : Modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of RecSys Conference*, 2011.

[10] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

[11] B. Ferwerda and M. Schedl. Enhancing music recommender systems with personality information and emotional states: A proposal. In *Proceedings of UMAP Conference*, 2014.

[12] M. Gillhofer and M. Schedl. Iron maiden while jogging, debussy for dinner? In X. He, S. Luo, D. Tao, C. Xu, J. Yang, and M. A. Hasan, editors, *MultiMedia Modeling: 21st International Conference, MMM 2015, Sydney, NSW, Australia, January 5-7, 2015, Proceedings, Part II*, pages 380–391, Cham, 2015. Springer International Publishing.

[13] S. Gopal, Y. Yang, B. Bai, and A. Niculescu-mizil. Bayesian models for large-scale hierarchical classification. In *Proceedings of NIPS*, 2012.

[14] N. Hariri, B. Mobasher, and R. Burke. Context-aware music recommendation based on latenttopic sequential patterns. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 131–138, New York, NY, USA, 2012. ACM.

[15] T. S. Jaakkola and M. I. Jordan. A variational approach to bayesian logistic regression models and their extensions. In *Workshop on Artificial Intelligence and Statistics*, 1996.

[16] D. Jannach, L. Lerche, and I. Kamehkhosh. Beyond "hitting the hits": Generating coherent music playlist continuations with the right tracks. In *Proceedings of the 9th ACM Conference on Recommender Systems*,

RecSys '15, pages 187–194, New York, NY, USA, 2015. ACM.

[17] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Journal of Machine Learning*, 1999.

[18] P. Knees, T. Pohle, M. Schedl, and G. Widmer. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. In *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, MIR '06, pages 147–154, New York, NY, USA, 2006. ACM.

[19] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[20] J. Lee. How similar is too similar ?: Exploring users ' perceptions of similarity in playlist evaluation. In *Proceedings of ISMIR Conference*, 2011.

[21] L. V. D. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008.

[22] D. J. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 1992.

[23] B. Mcfee, L. Barrington, and G. Lanckriet. Learning similarity from collaborative filters. In *Proceedings of ISMIR Conference*, 2010.

[24] B. Mcfee and G. Lanckriet. Heterogeneous embedding for subjective artist similarity. In *Proceedings of ISMIR Conference*, 2009.

[25] B. McFee and G. Lanckriet. Learning multi-modal similarity. *Journal of Machine Learning*, 2011.

[26] A. Mnih. Taxonomy-informed latent factor models for implicit feedback. In *JMLR W&CP Volume 18*, pages 169–181, 2012.

[27] U. Paquet and N. Koenigstein. One-class collaborative filtering with random graphs. In *Proceedings of WWW Conference*, 2013.

[28] S. Pauws. Pats: Realization and user evaluation of an automatic playlist generator. In *Proceedings of ISMIR Conference*, 2002.

[29] S. Pauws, W. Verhaegh, and M. Vossen. Music playlist generation by adapted simulated annealing. *Information Sciences*, 178(3):647 – 662, 2008. Including Special Issue âĂIJAmbient IntelligenceâĂİ.

[30] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 2000.

[31] M. Schedl and D. Hauger. Tailoring music recommendations to users by considering diversity, mainstreaminess, and novelty. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 947–950, New York, NY, USA, 2015. ACM.

[32] M. Slaney, K. Weinberger, and W. White. Learning a metric for music similarity. In *Proceedings of ISMIR Conference*, 2008.

[33] R. Turrin, M. Quadrana, A. Condorelli, R. Pagano, and P. Cremonesi. 30music listening and playlists dataset. In *Proceedings of RecSys Conference*, 2015.

[34] A. van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems 26 (2013)*. Proceedings of NIPS Conference, 2013.

[35] D. Zhou and L. Xiao. Hierarchical classification via orthogonal transfer. In *Proceedings of ICML Conference*, 2011.