Groove Radio: A Bayesian Hierarchical Model for Personalized Playlist Generation

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. 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 [16]. 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 [12, 22] 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 prediction task at hand. Beyond this evaluation, a short descrip-

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

tion of an online A/B experiment is provided, comparing a variant of the algorithm in this paper to Groove's previous playlist generation algorithm. The experiment shows an increase in user listening time and a reduction in the number of skipped tracks.

2. RELATED WORK

Automatic playlist generation is an active research field. The formulation of the playlist generation problem varies across literature and, consequently, various types of methods and evaluations are proposed [4]. This problem is mainly researched in the domains of Recommender Systems research and Music Information Retrieval (MIR).

Recommender systems excel at learning relations between users and items (personalization). Specifically, predicting music ratings in the presence of a genre taxonomy was the focus of the 2011 KDD-Cup competition [10]. In [9], a matrix factorization model was presented in which the domain taxonomy was utilized in order to propagate information between parameters with shared ancestors in the taxonomy (e.g. tracks belonging to the same artist etc). As explained earlier, this paper addresses a very different problem: finding the next track to be added to a playlist in the context of an artist seed, the user ID and previously played tracks. Unlike matrix factorization methods, it is not based on learning latent features. Instead, it builds upon multiple explicit information sources that include collaborative filtering along with meta-data, acoustic signals and popularity patterns. In essence, it is a hybrid approach [5] that builds upon similarities derived from various sources. It can also be viewed as a two-layered cascade in which different algorithms are utilized in the first layer to produce various types of similaritiy that serve as features to a second layer that produces the playlist. As such it bears resemblance to a recent paper on optimizing click-through rate in recommender systems [27].

In the music domain, similarities are typically derived from: usage [1, 20], meta-data [3, 21, 25, 28] or both [30]. Yet, producing good similarities between tracks is not always sufficient for generating high quality playlists [4]. Section 4 provides an overview of the features used in this paper. However, the focus of this work is on the underlying algorithm for generating the playlists given these similarities.

In MIR research, different approaches for playlist generation have been proposed. Typically, tracks are ranked based on audio features, meta-data and specific user queries [8, 17]. Learning from multiple types of similarity was investigated in [22], where the authors propose a method for integrating heterogeneous data into a single similarity space. Our method differs from [22] in several aspects: First, it is able to utilize the hierarchical semantic structure that exists in the taxonomy of the music domain. Second, it is formalized as a binary classification problem, whereas the method in [22] produces embedding in a latent space. Third, it contains a personalization component and generates playlists according to a specific combination of artist and user. Additionally, it provides a measure of confidence in the predictions inherent to Bayesian modeling.

Hierarchical classification models are well studied in the literature [6, 13, 31]. Hierarchical MIR models are proposed in [7, 9, 23] for the applications of genre classification, music recommendations, and artist organization, respectively. To the best of our knowledge, this paper is the first to introduce a Bayesian hierarchical model that utilizes both multiple sources and the domain taxonomy at various resolutions (genre, sub-genre and artist) for the application of automatic personalized playlist generation.

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. 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 $\sigma(x) \stackrel{\text{def}}{=} (1+\mathrm{e}^{-x})^{-1}$ 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 The Prior Distribution

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:

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

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

$$\boldsymbol{\theta} \stackrel{\text{def}}{=} \Big\{ \{\mathbf{w}^{(u)}_{k_u}\}_{k_u=1}^{U}, \{\mathbf{w}^{(a)}_{k_a}\}_{k_a=1}^{A}, \{\mathbf{w}^{(s)}_{k_s}\}_{k_s=1}^{S}, \{\mathbf{w}^{(g)}_{k_g}\}_{k_g=1}^{G}, \mathbf{w}, \tau_u, \tau_a, \tau_s, \tau_g, \tau_w \Big\},$$

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

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

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 $\mathcal{F}[q(\boldsymbol{\theta})] \stackrel{\text{def}}{=} \int q(\boldsymbol{\theta}) \log \frac{p(\mathcal{D}, \boldsymbol{\theta}|\mathcal{H})}{q(\boldsymbol{\theta})} \, \mathrm{d}\boldsymbol{\theta}$. $\mathcal{F}$ serves as a lower bound on the log marginal likelihood, or logarithm of the model evidence.

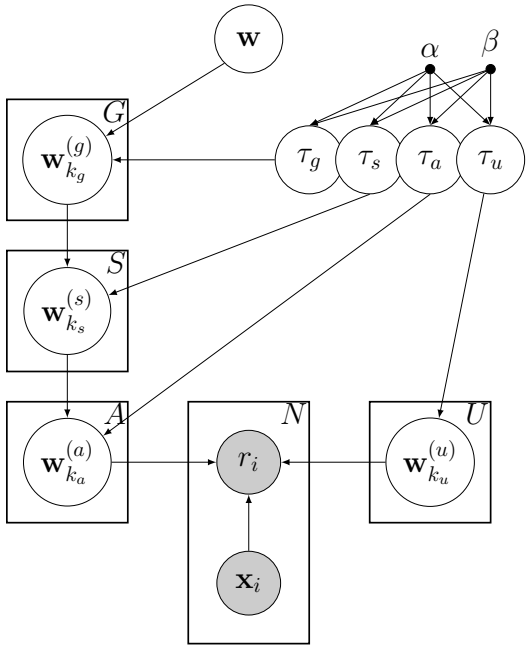

Figure 1: A graphical model representation of the proposed model. Unshaded circles denote unobserved random variables. Shaded circles denote observed random variables. Solid dots denote hyper-parameters.

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 [14].

First, the sigmoid functions appearing in (4) are lower-bounded using the logistic bound [15]. 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 $h_i \stackrel{\text{def}}{=} \mathbf{x}_i^\top(\mathbf{w}_{a_i}^{(a)} + \mathbf{w}_{u_i}^{(u)})$ and $\lambda_i \stackrel{\text{def}}{=} \frac{1}{2\xi_i}[\sigma(\xi_i) - \frac{1}{2}]$. 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:

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

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

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

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

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

$$\boldsymbol{\Sigma}_{k_u}^{(u)} = \left[\tau_u \mathbf{I} + \sum_{i=1}^{N} \mathbb{I}\left[u_i = k_u\right] 2\lambda_i \cdot \mathbf{x}_i \mathbf{x}_i^\top\right]^{-1}$$

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

where $\mathbb{I}[\cdot]$ 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 $k_a = 1 \ldots A$ we approximate the posterior of $\mathbf{w}_{k_a}^{(a)}$ with a Normal distribution

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

$$\boldsymbol{\Sigma}_{k_a}^{(a)} = \left[\tau_a \cdot \mathbf{I} + \sum_{i=1}^{N} \mathbb{I}\left[a_i = k_a\right] 2\lambda_i \cdot \mathbf{x}_i \mathbf{x}_i^\top\right]^{-1},$$

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

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

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

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

where $\mathcal{C}_{k_s} = \mathtt{child}(k_s)$ 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 $\mathbf{w}_{k_g}^{(g)}$ with a Normal distribution

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

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

where $\mathcal{C}_{k_g} = \mathtt{child}(k_g)$ 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

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

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

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 $\beta_u = \beta + \frac{1}{2}\sum_{k_u=1}^{U} \langle \left(\mathbf{w}_{k_u}^{(u)}\right)^\top \mathbf{w}_{k_u}^{(u)} \rangle$ 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 $\xi_i = \sqrt{\langle \left(\mathbf{x}_i^\top \left(\mathbf{w}_{a_i}^{(a)} + \mathbf{w}_{u_i}^{(u)}\right)\right)^2 \rangle}$. 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:

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

where the random variable h_m has a Normal distribution:

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

Finally, the integral in (12) is approximated following MacKay [19] 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. We divide the features into four groups categorized according to the type of signal employed in their calculation:

Acoustic Audio Features (AAF) - Audio similarity between two tracks is computed following [26], by extracting Mel-Frequency Cepstral Coefficients (MFCCs) from a track's audio samples and learning a Gaussian Mixture Model (GMM) representation of their distribution . Track similarity is achieved by estimating the Kullback-Leibler divergence between the coefficient distributions of two tracks. To compute similarities at the artist level, a similar approach is applied by fitting GMM models to audio samples of multiple tracks by the same artist.

Usage Features (UF) - These features are derived using collaborative filtering techniques on music consumption signals. Similarities are computed following [24], by learning a low-rank matrix factorization of the user-track and user-artist implicit usage signals. The output of this process is a vector representation for each track and artist. Similarities are measured using vector space distances.

Meta-Data Features (MDF) - These features are based on the meta-data tags (e.g. "easy-listening", "upbeat", "90's"). Similarities are computed by considering each track/artist as a sparse binary vector where each entry represents the existence of a semantic tag. The cosine similarity of these vectors provides the similarity measure applied between the candidate track and previous tracks as well as between a candidate track's artist and the seed.

Popularity Features (PF) - Popularity is a measure of the prevalence of a track or artist in the dataset. First, it is used to compute unary features representing the popularity of a candidate track and its artist. Second, pairwise features are computed relating the popularity of the candidate track and its artist to the popularity of the seed artist and previous tracks.

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 midplay by the user.

30Music Dataset is a publicly available dataset of user playlists [29]. 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 [11]. 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 80% of the examples and evaluated on the remaining 20%. We compare it against several baselines:

Partial Taxonomy - This baseline learns only a subset of the domain taxonomy parameters, corresponding to a higher level in the hierarchy than the full model. Evaluations are provided at each of the sub-hierarchies: global (no hierarchy), genre , and sub-genre.

Non-Personalized - A non-personalized version of the Groove model is evaluated by removing the per-user parameters.

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 model vs. the non-personalized baseline at different levels of the hierarchical taxonomy. The effect of adding personalization is apparent in both 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.

	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

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 3 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 4 provides an illustration of artist parameters using a t-SNE embedding [18] 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 Online A/B Testing

We performed an A/B experiment comparing a variant of the model described above with Groove's previous playlist algorithm. The experiment was conducted by randomly assigning users to buckets and measuring online metrics designed to quantify user satisfaction and engagement. Specifically, we consider *Average Session Length* (ASL), and *User Average Skip Ratio* (UASR). ASL is the listening time elapsed

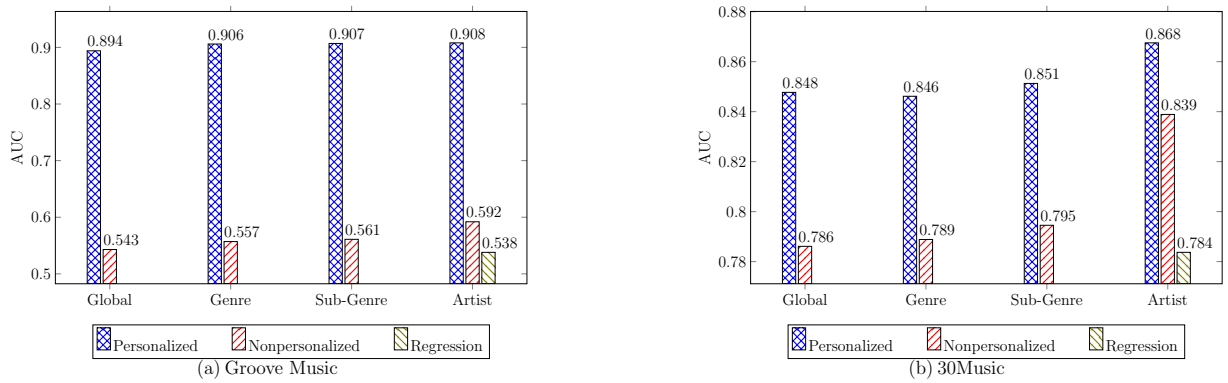

Figure 2: The effects of personalization and hierarchy on accuracy (as measured by AUC) are illustrated for (a) the Groove Music dataset (a) the 30Music Dataset. We evaluate the personalized and non-personalized approach at each level of the partial hierarchy (increasing depth from left to right). The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

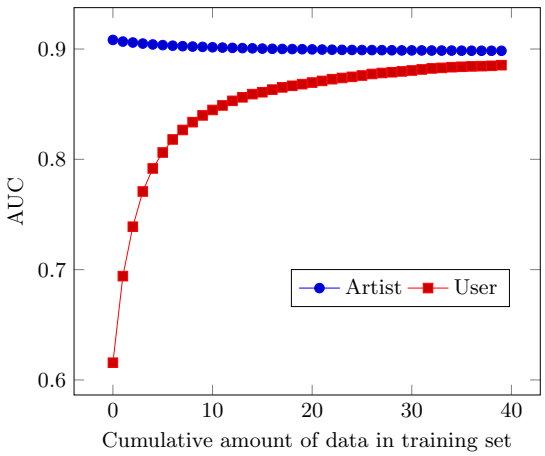

Figure 3: A cumulative plot illustrating the effect of Artist/User coldness on test AUC. The AUC is plotted for all Artists/Users with at least x labeled data points in the training set.

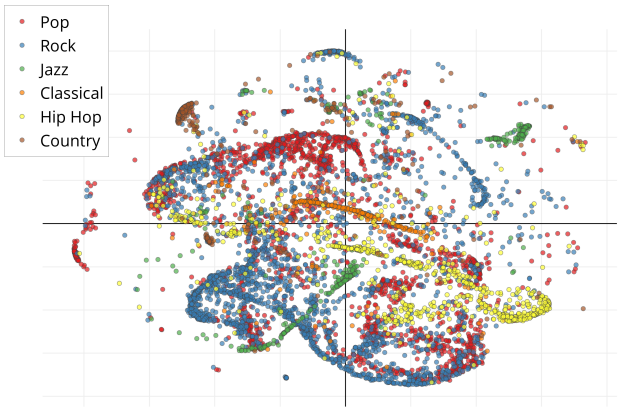

Figure 4: tSNE embedding of artist parameters. The figure shows a complex manifold that could not be captured by modeling higher levels of the taxonomy alone.

in a single radio session, averaged over all such sessions. UASR is a the ratio of skipped tracks to tracks that were played to completion averaged over all users. The approach in this paper was able to outperform the baseline on both metrics showing an increase of 4.21% in ASL and a 5.76% decrease in UASR.

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 $m = 1 \dots M$ 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] R. Burke. Hybrid recommender systems: Survey and experiments. In *User Modeling and User-Adapted Interaction*, 2002.

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

[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] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The Yahoo! music dataset and KDD-Cup'11. In *Proceedings of KDD Conference*, 2011.

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

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

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

[14] 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.

[15] 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.

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

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

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

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

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

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

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

[23] E. Pampalk, A. Flexer, and G. Widmer. Hierarchical organization and description of music collections at the artist level. In *Proceedings of ECDL*, 2005.

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

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

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

[27] O. Sar-Shalom, N. Koenigstein, U. Paquet, and H. P. Vanchinathan. Beyond collaborative filtering: The list recommendation problem. In *Proceedings of WWW Conference*, 2016.

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

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

[30] 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.

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