Time Series Models for Semantic Music Annotation

Emanuele Coviello, Antoni B. Chan, and Gert Lanckriet

***Abstract*—Many state-of-the-art systems for automatic music tagging model music based on bag-of-features representations which give little or no account of temporal dynamics, a key characteristic of the audio signal. We describe a novel approach to automatic music annotation and retrieval that captures temporal (e.g., rhythmical) aspects as well as timbral content. The proposed approach leverages a recently proposed song model that is based on a generative time series model of the musical content—the dynamic texture mixture (DTM) model—that treats fragments of audio as the output of a linear dynamical system. To model characteristic temporal dynamics and timbral content at the tag level, a novel, efficient, and hierarchical expectation–maximization (EM) algorithm for DTM (HEM-DTM) is used to summarize the common information shared by DTMs modeling individual songs associated with a tag. Experiments show learning the semantics of music benefits from modeling temporal dynamics.**

***Index Terms*—Audio annotation and retrieval, dynamic texture model, music information retrieval.**

I. Introduction

RECENT technologies fueled new trends in music production, distribution, and sharing. As a consequence, an already large corpus of millions of musical pieces is constantly enriched with new songs (by established artists as well as less known performers), all of which are instantly available to millions of consumers through online distribution channels, personal listening devices, etc. This age of music proliferation created a strong need for music search and discovery engines, to help users find "Mellow Beatles songs" on a nostalgic night, or satisfy their sudden desire for "psychedelic rock with distorted guitar and deep male vocals," without knowing appropriate artists or song titles. A key scientific challenge in creating this search technology is the development of intelligent algorithms, trained to map the human perception of music within the coded confine of computers, to assist in automatically analyzing, indexing and recommending from this extensive corpus of musical content [1].

This paper concerns *automatic tagging* of music with descriptive keywords (e.g., genres, emotions, instruments, usages, etc.), based on the content of the song. Music annotations can be used for a variety of purposes, such as searching for songs exhibiting specific qualities (e.g., "jazz songs with female vocals and saxophone"), or retrieval of semantically similar songs (e.g., generating play-lists based on songs with similar annotations). Since semantics is a compact, popular medium to describe an auditory experience, it is essential that a music search and discovery system supports these semantics-based retrieval mechanisms, to recommend content from a large audio database.

State-of-the-art music "auto-taggers" represent a song as a "bag of audio features" (e.g., [2]–[6]). The bag-of-features representation extracts audio features from the song at regular time intervals, but then treats these features independently, ignoring the temporal order or dynamics between them. Hence, this representation fails to account for the longer-term musical dynamics (e.g., tempo and beat) or temporal structures (e.g., riffs and arpeggios), which are clearly important characteristics of a musical signal.

To address this limitation, one approach is to encode some temporal information in the features ([2], [4]–[8]) and keep using existing, time-independent models. For example, some of the previous approaches augment the "bag of audio features" representation with the audio features' first and second derivatives. While this can slightly enrich the representation at a short-time scale, it is clear that a more principled approach is required to model dynamics at a longer-term scale (seconds instead of milliseconds).

Therefore, in this paper, we explore the dynamic texture (DT) model [9], a generative *time series model* that captures longer-term time dependencies, for automatic tagging of musical content. The DT model represents a time series of audio features as a sample from a *linear dynamical system* (LDS), which is similar to the hidden Markov model (HMM) that has proven robust in music identification [10]. The difference is that HMMs quantize the audio signal into a fixed number of discrete "phonemes," while the DT has a continuous state space, offering a more flexible model for music.

Since musical time series often show significant structural changes within a single song and have dynamics that are only locally homogeneous, a single DT would be insufficiently rich to model individual songs and, therefore, the typical musical content associated with semantic tags. To address this at the song-level, Barrington *et al.* [11] propose to model the audio fragments from a *single song* as samples from a dynamic texture mixture (DTM) model [12], for the task of automatic music *segmentation*. Their results demonstrated that the DTM provides an accurate segmentation of music into homogeneous, perceptually similar segments (corresponding to what a human listener would label as "chorus," "verse," "bridge," etc.) by capturing *temporal* as well as *textural* aspects of the musical signal.

In this paper, we adopt the DTM model to propose a novel approach to the task of automatic music *annotation* that accounts for both the timbral content and the temporal dynamics

Manuscript received August 08, 2010; revised October 11, 2010; accepted October 15, 2010. Date of publication October 28, 2010; date of current version May 13, 2011. The works of E. Coviello and G. Lanckriet were supported by the National Science Foundation under Grants DMS-MSPA 0625409 and CCF-0830535. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Bryan Pardo.

E. Coviello and G. Lanckriet are with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093 USA (e-mail: ecoviell@ucsd.edu; gert@ece.ucsd.edu).

A. B. Chan is with the Department of Computer Science, City University of Hong Kong, Hong Kong, China (e-mail: abchan@cityu.edu.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASL.2010.2090148

that are predictive of a semantic *tag*. We first model each song in a music database as a DTM, capturing longer-term time dependencies and instantaneous spectral content at the *song-level*. Second, the characteristic temporal and timbral aspects of musical content commonly associated with a semantic tag are identified by learning a *tag-level* DTM that summarizes the common features of a (potentially large) set of song-level DTMs for the tag (as opposed to the tag-level Gaussian mixture models by Turnbull *et al.* [2], which do not capture temporal dynamics). Given all song-level DTMs associated with a particular tag, the common information is summarized by clustering similar song-level DTs using a novel, efficient hierarchical EM (HEM-DTM) algorithm. This gives rise to a tag-level DTM with few mixture components.

Experimental results show that the proposed time series model improves annotation and retrieval, in particular for tags with temporal dynamics that unfold in the time span of a few seconds.

In summary, this paper brings together a DTM model for music, a generative framework for music annotation and retrieval, and an efficient HEM-DTM algorithm. We will focus our discussion on the latter two. For the former, we provide an introduction and refer to our earlier work [11] for more details. The remainder of this paper is organized as follows. After an overview of related work on auto-tagging of music in Section II, we introduce the DTM model in Section III. Next, in Sections IV and V, we present an annotation and retrieval system for time series data, based on an efficient hierarchical EM algorithm for dynamic texture mixtures (HEM-DTM). In Sections VI and VII, we present experiments using HEM-DTM for music annotation and retrieval. Finally, Section VIII illustrates qualitatively how variations in the acoustic characteristics of semantic tags affect the parameters of the corresponding DTM models.

II. Related Work

The prohibitive cost of manual labeling makes automated semantic understanding of audio content a core challenge in designing fully functional retrieval systems ([2]–[8], [13]–[22]). To automatically annotate music with *semantic tags*, based on audio content, various discriminative machine learning algorithms have been proposed (e.g., multiple-instance [5], multiple-kernel [17], and stacked [3] support vector machines (SVMs), boosting [6], nearest-neighbor ([18], [19]), embedding methods [20], locally sensitive hashing [7] and regularized least-squares [22]). The discriminative framework, however, can suffer from poorly or weakly labeled training data (e.g., positive examples considered as negatives due to incomplete annotations).

To overcome this problem, unsupervised learning algorithms have been considered (e.g., K-means [23], vector quantization [10]), ignoring any labels and determining the classes automatically. The learned clusters, however, are not guaranteed to have any connection with the underlying semantic tags of interest.

The labeling problem is compounded since often only a subset of the song's features actually manifests the tag the entire song is labeled with (e.g., a song labeled with "saxophone" may only have a handful of features describing content where a saxophone is playing). This suggests a *generative modeling* approach, which is better suited at handling weakly labeled data and estimating concept distributions that naturally emerge around concept-relevant audio content, while down-weighting irrelevant outliers. More details on how generative models accommodate weakly labeled data by taking a multiple instance learning approach is provided by Carneiro *et al.* [24]. Moreover, generative models provide class-conditional probabilities, which naturally allows us to rank tags probabilistically for a song. Generative models have been applied to various music information retrieval problems. This includes Gaussian mixture models (GMMs) ([2], [21], [25]), hidden Markov models (HMMs) [10], hierarchical Dirichlet processes (HDPs) [26], and a codeword Bernoulli average model (CBA) [4]. Generative models used for automatic music annotation (e.g., GMMs and CBA) usually model the spectral content (and, sometimes, its first and second instantaneous derivatives) of short-time windows. These models ignore longer-term temporal dynamics of the musical signal. In this paper, we adopt dynamic texture mixture models for automatic music annotation. These generative time-series models capture both instantaneous spectral content, as well as longer-term temporal dynamics. Compared to HMMs, they have a continuous rather than discrete state space. Therefore, they do not require to quantize the rich sound of a musical signal into discrete "phonemes," making them an attractive model for music.

III. Dynamic Texture Mixture Models

In this section, we review the dynamic texture (DT) and dynamic texture mixture (DTM) models for modeling short audio fragments and whole songs.

A. Dynamic Texture Model

A DT [9] is a generative model that takes into account both the instantaneous acoustics and the temporal dynamics of audio sequences (or audio fragments) [11]. The model consists of two random variables: y_t, which encodes the acoustic component (audio feature vector) at time t, and x_t, a hidden state variable which encodes the dynamics (evolution) of the acoustic component over time. The two variables are modeled as a *linear dynamical system*:

$$x_t = Ax_{t-1} + v_t,$$
$$y_t = Cx_t + w_t + \bar{y}$$

where $x_t \in \mathbb{R}^n$ and $y_t \in \mathbb{R}^m$ are real vectors (typically $n \ll m$). Using such a model, we assume that the dynamics of the audio can be summarized by a more parsimonious *hidden state process* $x_t (n < m)$, which evolves as a first order Gauss–Markov process, and each *observation variable* y_t is dependent only on the current hidden state x_t.

The *state transition matrix* $A \in \mathbb{R}^{n \times n}$ encodes the dynamics or evolution of the hidden state variable (e.g., the evolution of the audio track), and the *observation matrix* $C \in \mathbb{R}^{m \times n}$ encodes the basis functions for representing the audio fragment. The vector $\bar{y} \in \mathbb{R}^m$ is the mean of the dynamic texture (i.e., the mean audio feature vector). The *driving noise process* v_t is zero-mean Gaussian distributed with covariance Q, i.e., $v_t \sim \mathcal{N}(0, Q)$, with $Q \in \mathbb{S}_+^n$, the set of symmetric, positive definite matrices of dimension $n \times n$. w_t is the *observation noise* and is also

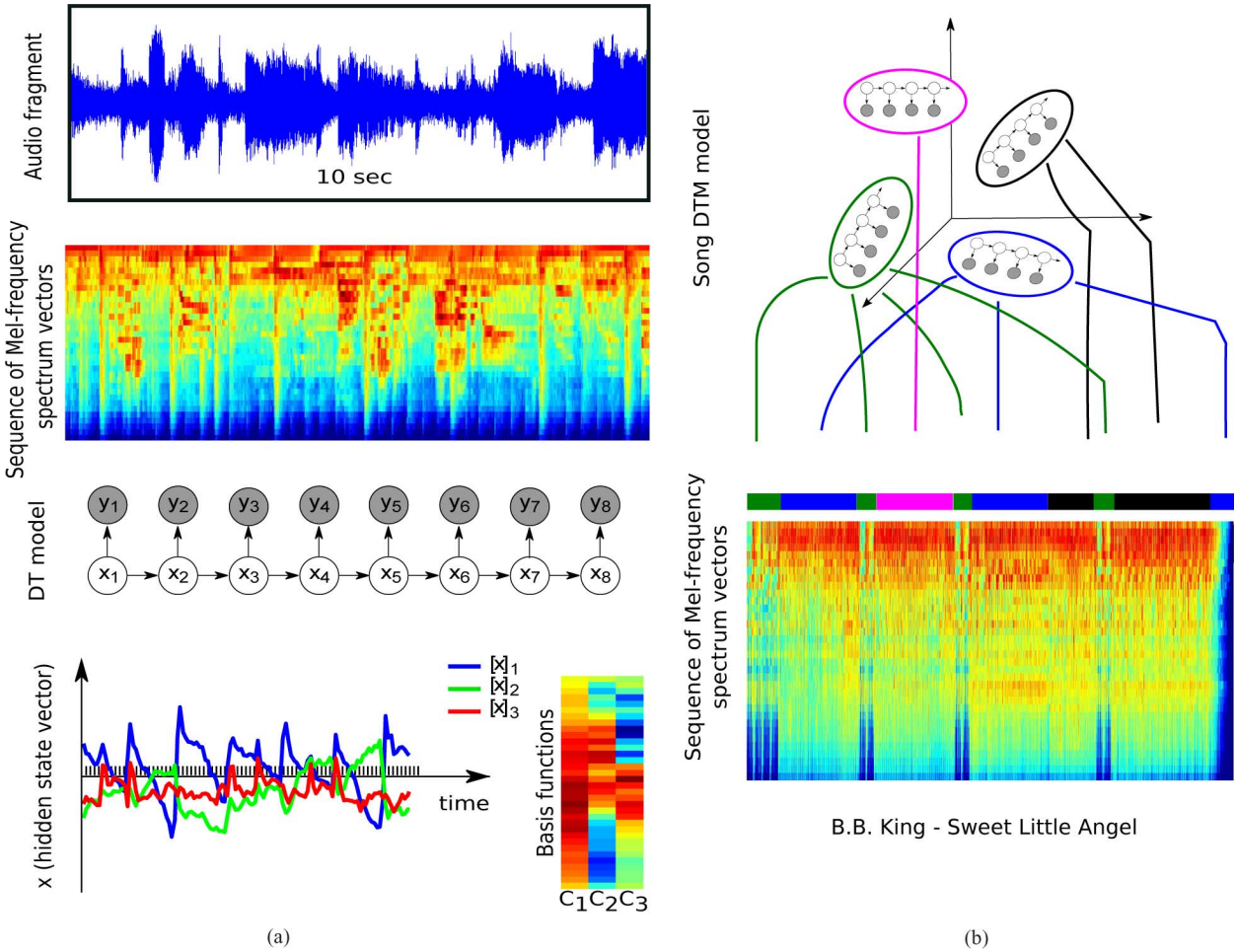

Fig. 1. Dynamic texture music model. (a) A single DT represents a short audio fragment. (b)A DT mixture represents the heterogeneous structure of a song, with individual mixture components modeling homogeneous sections. The different orientations (and, locations) of the DT components in the top part of (b) are to visually suggest that each DT is characterized by a distinct set of parameters, to produce a specific type of audio fragments. (a) DT model. (a) DTM model.

zero-mean Gaussian, with covariance R, i.e., $w_t \sim \mathcal{N}(0, R)$, with $R \in \mathbb{S}_+^m$. Finally, the *initial condition* is distributed as $x_1 \sim \mathcal{N}(\mu, S)$, where $\mu \in \mathbb{R}^n$ is the mean of the initial state, and $S \in \mathbb{S}_+^n$ the covariance. The DT is specified by the parameters $\Theta = \{A, Q, C, R, \mu, S, \bar{y}\}$.

Intuitively, the columns of C can be interpreted as the principal components (or basis functions) of the audio feature vectors over time. Hence, each audio feature vector y_t can be represented as a linear combination of principal components, with corresponding weights given by the current hidden state x_t. In this way, the DT can be interpreted as a time-varying PCA representation of an audio feature vector time series. Fig. 1(a) shows the graphical model of the DT, as it represents a short audio fragment.

B. Dynamic Texture Mixture Model

A song is a combination of heterogeneous audio fragments with significant structural variations, and hence cannot be represented with a single DT model. To address this lack of global homogeneity, Barrington *et al.* [11] proposed to represent audio fragments, extracted from a song, as samples from a dynamic texture mixture (DTM) [12], effectively modeling the heterogeneous structure of the song. The DTM model [12] introduces an assignment random variable $z \sim \text{multinomial}(\pi_1, \ldots, \pi_K)$, which selects one of K dynamic texture components as the source of an audio fragment. Each mixture component is parameterized by

$$\Theta_z = \{A_z, C_z, Q_z, R_z, \mu_z, S_z, \bar{y}_z\} \tag{1}$$

and the DTM model is parameterized by $\Theta = \{\pi_z, \Theta_z\}_{z=1}^K$.

Given a set of audio fragments extracted from a song, the maximum-likelihood parameters of the DTM can be estimated with recourse to the expectation–maximization (EM) algorithm, which is an iterative optimization method that alternates between estimating the hidden variables with the current parameters, and computing new parameters given the estimated hidden variables (the "complete data"). The EM algorithm for DTM alternates between estimating second-order statistics of the hidden states, conditioned on each audio fragment, with the Kalman smoothing filter (E-step), and computing new parameters given these statistics (M-step). More details are provided by Chan and Vasconcelos [12].

Fig. 1(b) illustrates the DTM representation of a song, where each DT component models homogeneous parts of the song. Previous work by Barrington *et al.* [11] has successfully used the DTM for the task of segmenting the structure of a song into acoustically similar sections (e.g., intro, verse, chorus, bridge,

solo, outro). In this paper, we propose that the DTM can also be used as a *tag-level annotation model* for music annotation and retrieval.

IV. Music Annotation and Retrieval With DTMS

In this section, we formulate the related tasks of annotation and retrieval of audio data as a supervised multi-class labeling (SML) problem [24] in the context of time series DTM models.

A. Notation

A song $\mathcal{Y}$ is represented as a collection of T overlapping time series, i.e., $\mathcal{Y} = \{y_{1:\tau}^1, \ldots, y_{1:\tau}^T\}$, where each $y_{1:\tau}^t$, called an audio *fragment*, represents τ sequential audio feature vectors extracted by passing a short-time window over the audio signal. The number of audio fragments, T, depends on the length of the song. The semantic content of a song with respect to a vocabulary $\mathcal{V}$ of size $|\mathcal{V}|$ is represented in an annotation vector $\mathbf{c} = [c_1, \ldots, c_{|\mathcal{V}|}]$, where $c_k > 0$ only if there is a positive association between the song and the tag w_k, otherwise $c_k = 0$. Each *semantic weight* c_k represents the degree of association between the song and the tag w_k. The data set $\mathcal{D}$ is a collection of $|\mathcal{D}|$ song-annotation pairs $(\mathcal{Y}_d, \mathbf{c}_d)$.

B. Music Annotation

We treat annotation as a supervised multi-class problem [2], [24] in which each class is a tag w, from a vocabulary $\mathcal{V}$ of unique tags (e.g., "bass guitar," "hip hop," "boring"). Each tag w_k is modeled with a probability distribution over the space of audio fragments, i.e., $p(y_{1:\tau}^t|w_k)$ for $k = 1, \ldots, |\mathcal{V}|$, which is a DTM. The annotation task is to find the subset $\mathcal{W} = \{w_1, \ldots, w_{\mathcal{A}}\} \subseteq \mathcal{V}$ of $\mathcal{A}$ tags that best describe a novel song $\mathcal{Y}$.

Given the audio fragments of a novel song $\mathcal{Y}$, the most relevant tags are the ones with highest posterior probability, computed using Bayes' rule:

$$p(w_k|\mathcal{Y}) = \frac{p(\mathcal{Y}|w_k)p(w_k)}{p(\mathcal{Y})} \tag{2}$$

where $p(w_k)$ is the prior of the kth tag and $p(\mathcal{Y})$ the song prior. To promote annotation using a diverse set of tags, we assume a uniform prior, i.e., $p(w_k) = 1/|\mathcal{V}|$ for $k = 1, \ldots, |\mathcal{V}|$. To estimate the likelihood term in (2), $p(\mathcal{Y}|w_k)$, we assume that song fragments $y_{1:\tau}^t$ are conditionally independent (given w_k). To compensate for the inaccuracy of this naïve Bayes assumption and keep the posterior from being too "peaked," one common solution is to estimate the likelihood term with the geometric average [2] (in this case, the geometric average of the individual audio fragment likelihoods):

$$p(\mathcal{Y}|w_k) = \prod_{t=1}^{T} \left(p\left(y_{1:\tau}^t|w_k\right)\right)^{\frac{1}{T\tau}}. \tag{3}$$

Note that, besides normalizing by T, we also normalize by the length of the audio fragment, τ, due to the high dimension of the probability distribution of the DTM time series model. The likelihood terms $p(y_{1:\tau}^t|w_k)$ of the DTM tag models can be computed efficiently with the "innovations" form of the likelihood using the Kalman filter [12], [27].

Unlike bag-of-features models that discard any dependency between audio feature vectors, (3) only assumes independence between different *sequences* of audio feature vectors (i.e., audio fragments, describing seconds of audio). Correlation *within* a single sequence is directly accounted for by the time series model.

The probability that the song $\mathcal{Y}$ can be described by the tag w_k is

$$p(w_k|\mathcal{Y}) = \frac{\prod_{t=1}^{T} \left(p\left(y_{1:\tau}^t|w_k\right)\right)^{\frac{1}{T\tau}}}{\sum_{l=1}^{|\mathcal{V}|} \prod_{t=1}^{T} \left(p\left(y_{1:\tau}^t|w_l\right)\right)^{\frac{1}{T\tau}}} \tag{4}$$

where the song prior $p(\mathcal{Y}) = \sum_{l=1}^{|\mathcal{V}|} p(\mathcal{Y}|w_l)p(w_l)$. Finally, the song can be represented as a semantic multinomial, $\mathbf{p} = [p_1, \ldots, p_{|\mathcal{V}|}]$, where each $p_k = p(w_k|\mathcal{Y})$ represents the relevance of the kth tag for the song, and $\sum_{i=1}^{|\mathcal{V}|} p_i = 1$. We annotate a song with the most likely tags according to $\mathbf{p}$, i.e., we select the tags with the largest probability.

C. Music Retrieval

Given a tag-based query, songs in the database can be retrieved based on their relevance to this semantic query.[1] In particular, we determine a song's relevance to a query with tag w_k based on the posterior probability of the tag, $p(w_k|\mathcal{Y})$, in (4). Hence, retrieval involves rank-ordering the songs in the database, based on the kth entry (p_k) of the semantic multinomials $\mathbf{p}$.

Note that the songs could also be ranked by the likelihood of the song given the query, i.e., $p(\mathcal{Y}|w_k)$. However, this tends not to work well in practice because it favors generic songs that are most similar to the song prior $p(\mathcal{Y})$, resulting in the same retrieval result for any query w_k. Normalizing by the song prior $p(\mathcal{Y})$ fixes this problem, yielding the ranking based on semantic multinomials (assuming a uniform tag prior) described above.

V. Learning DTM Tag Models With the Hierarchical EM Algorithm

In this paper, we represent the tag models with dynamic texture mixture models. In other words, the tag distribution $p(y_{1:\tau}^t|w_k)$ is modeled with the probability density of the DTM, which is estimated from the set of training songs associated with the particular tag. One approach to estimation is to extract all the audio fragments from the relevant training songs, and then run the EM algorithm [12] directly on this data to learn the tag-level DTM. This approach, however, requires storing many audio fragments in memory (RAM) for running the EM algorithm. For even modest-sized databases, the memory requirements can exceed the RAM capacity of most computers.

To allow efficient training in both computation time and memory requirements, the learning procedure is split into two steps. First, a song-level DTM model is learned for each song in the training set using the standard EM algorithm [12]. Next, a tag-level model is formed by pooling together all the song-level DTMs associated with a tag, to form a large mixture. However, a drawback of this model aggregation approach is that the number of DTs in the DTM tag model grows linearly with the

[1]Note that although this work focuses on single-tag queries, our representation easily extends to multiple-tag queries [28].

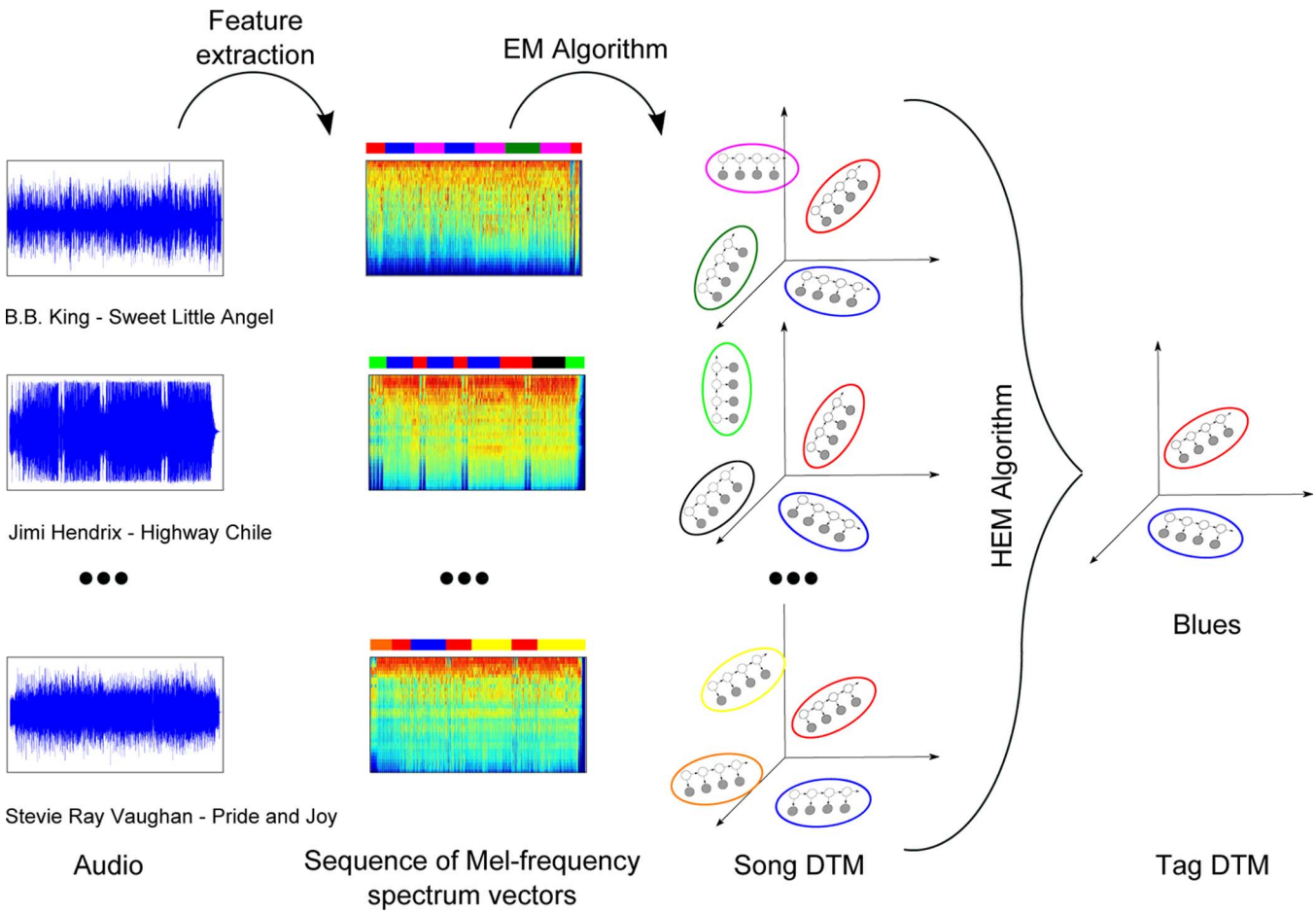

Fig. 2. Learning a DTM tag model: first song-level DTMs are learned with EM for all songs associated with a tag, e.g., "Blues." Then, the song-level models are aggregated using HEM to find common features between the songs.

size of the training data, making inference computationally inefficient when using large training sets. To alleviate this problem, the DTM tag models formed by model aggregation are reduced to a representative DTM with fewer components by using the hierarchical EM (HEM) algorithm presented in this section. The HEM algorithm clusters together similar DTs in the song-level DTMs, thus summarizing the common information in songs associated with a particular tag. The new DTM tag model allows for more efficient inference, due to fewer mixture components, while maintaining a reliable representation of the tag-level model.

Because the database is first processed at the song level, the computation can be easily done in parallel (over the songs) and the memory requirement is greatly reduced to that of processing a single song. The memory requirement for computing the tag-level models is also reduced, since each song is succinctly modeled by the parameters of a DTM. Such a reduction in computational complexity also ensures that the tag-level models can be learned from cheaper, weakly labeled data (i.e., missing labels, labels without segmentation data) by pooling over large amounts of audio data to amplify the appropriate attributes.

In summary, adopting DTM, or time series models in general, as a tag model for SML annotation requires an appropriate HEM algorithm for efficiently learning the tag-level models from the song-level models. In the remainder of the section, we present the HEM algorithm for DTM.

A. *Learning DTM Tag Models*

The process for learning a tag-level DTM model from song-level DTMs is illustrated in Fig. 2. First, all the song-level DTMs with a particular tag are pooled together into a single large DTM. Next, the common information is summarized by clustering similar DT components together, forming a new *tag-level* DTM with fewer mixture components.

The DT components are clustered using the hierarchical expectation–maximization (HEM) algorithm [29]. At a high level, this is done by generating *virtual samples* from each of the song-level component models, merging all the samples, and then running the standard EM algorithm on the merged samples to form the reduced tag-level mixture. Mathematically, however, using the virtual samples is equivalent to marginalizing over the distribution of song-level models. Hence, the tag model can be learned directly and efficiently from the parameters of the song-level models, without generating any virtual samples.

The HEM algorithm was originally proposed by Vasconcelos and Lippman [29] to reduce a Gaussian mixture model (GMM) with a large number of mixture components into a representative GMM with fewer components, and has been successful in

learning GMMs from large datasets for the annotation and retrieval of images [24] and music [2]. We next present an HEM algorithm for mixtures with components that are *dynamic textures* [30].

B. HEM Formulation

Formally, let $\Theta^{(s)} = \{\pi_i^{(s)}, \Theta_i^{(s)}\}_{i=1}^{K^{(s)}}$ denote the combined *song-level* DTM (i.e., after pooling all song-level DTMs for a certain tag) with $K^{(s)}$ components, where $\Theta_i^{(s)}$ are the parameters for the ith DT component, and $\pi_i^{(s)}$ the corresponding component weights, which are normalized to sum to 1 (i.e., $\sum_i \pi_i^{(s)} = 1$). The likelihood of observing an audio fragment $y_{1:\tau}$ with length τ from the combined song-level DTM $\Theta^{(s)}$ is given by

$$p\left(y_{1:\tau}|\Theta^{(s)}\right) = \sum_{i=1}^{K^{(s)}} \pi_i^{(s)} p\left(y_{1:\tau}|z^{(s)} = i, \Theta^{(s)}\right) \quad (5)$$

where $z^{(s)} \sim \text{multinomial}(\pi_1^{(s)}, \ldots \pi_{K^{(s)}}^{(s)})$ is the hidden variable that indexes the mixture components. $p(y_{1:\tau}|z^{(s)} = i, \Theta^{(s)})$ is the likelihood of the audio fragment $y_{1:\tau}$ under the ith DT mixture component.

The goal is to find a tag-level *annotation* DTM, $\Theta^{(a)} = \{\pi_j^{(a)}, \Theta_j^{(a)}\}_{j=1}^{K^{(a)}}$, which represents (5) using fewer number of mixture components, $K^{(a)}$, (i.e., $K^{(a)} < K^{(s)}$). The likelihood of observing an audio fragment $y_{1:\tau}$ from the tag-level DTM $\Theta^{(a)}$ is

$$p\left(y_{1:\tau}|\Theta^{(a)}\right) = \sum_{j=1}^{K^{(a)}} \pi_j^{(a)} p\left(y_{1:\tau}|z^{(a)} = j, \Theta^{(a)}\right) \quad (6)$$

where $z^{(a)} \sim \text{multinomial}(\pi_1^{(a)}, \ldots, \pi_{K^{(a)}}^{(a)})$ is the hidden variable for indexing components in $\Theta^{(a)}$. Note that we will always use i and j to index the components of the song-level model $\Theta^{(s)}$ and the tag-level model $\Theta^{(a)}$, respectively. To reduce clutter, we will also use the short-hand $\Theta_i^{(s)}$ and $\Theta_j^{(a)}$ to denote the ith component of $\Theta^{(s)}$ and the jth component of $\Theta^{(a)}$, respectively. For example, we denote $p(y_{1:\tau}|z^{(s)} = i, \Theta^{(s)})$ as $p(y_{1:\tau}|\Theta_i^{(s)})$.

C. Parameter Estimation

To obtain the tag-level model, HEM [29] considers a set of N virtual observations drawn from the song-level model $\Theta^{(s)}$, such that $N_i = N\pi_i^{(s)}$ samples are drawn from the ith component. We denote the set of N_i virtual audio samples for the ith component as $Y_i = \{y_{1:\tau}^{(i,m)}\}_{m=1}^{N_i}$, where $y_{1:\tau}{}^{(i,m)} \sim \Theta_i^{(s)}$ is a single audio sample and τ is the length of the virtual audio samples (a parameter we can choose). The entire set of N samples is denoted as $Y = \{Y_i\}_{i=1}^{K^{(s)}}$. To obtain a consistent hierarchical clustering, we also assume that all the samples in a set Y_i are eventually assigned to the same tag-level component $\Theta_j^{(a)}$. We denote this as $z_i^{(a)} = j$. The parameters of the tag-level model can then be estimated by maximizing the likelihood of the virtual audio samples

$$\Theta^{(a)*} = \arg\max_{\Theta^{(a)}} \log p(Y|\Theta^{(a)}) \quad (7)$$

where

$$\begin{aligned}
&\log p(Y|\Theta^{(a)}) \\
&\quad = \log \prod_{i=1}^{K^{(s)}} p(Y_i|\Theta^{(a)}) \quad (8)\\
&\quad = \log \prod_{i=1}^{K^{(s)}} \sum_{j=1}^{K^{(a)}} \pi_j^{(a)} \int p\left(Y_i, X_i|z_i^{(a)} = j, \Theta^{(a)}\right) dX_i \\
&\quad = \log \prod_{i=1}^{K^{(s)}} \sum_{j=1}^{K^{(a)}} \pi_j^{(a)} \int p\left(Y_i, X_i|\Theta_j^{(a)}\right) dX_i \quad (9)
\end{aligned}$$

and $X_i = \{x_{1:\tau}^{(i,m)}\}_{m=1}^{N_i}$ are the hidden state variables corresponding to Y_i. Computing the log-likelihood in (9) requires marginalizing over the hidden assignment variables $z_i^{(a)}$ and hidden state variables X_i. Hence, (7) can also be solved with recourse to the EM algorithm [31]. In particular, each iteration consists of

$$\begin{aligned}
&\text{E–Step}: \quad \mathcal{Q}\left(\Theta^{(a)}, \hat{\Theta}^{(a)}\right) \\
&\qquad = \mathbb{E}_{X,Z|Y,\hat{\Theta}^{(a)}}\left[\log p\left(X, Y, Z|\Theta^{(a)}\right)\right] \\
&\text{M–Step}: \quad \hat{\Theta}^{(a)*} \\
&\qquad = \arg\max_{\Theta^{(a)}} \mathcal{Q}(\Theta^{(a)}, \hat{\Theta}^{(a)})
\end{aligned}$$

where $\hat{\Theta}^{(a)}$ is the current estimate of the tag-level model, $p(X, Y, Z|\Theta^{(a)})$ is the "complete-data" likelihood, and $\mathbb{E}_{X,Z|Y,\hat{\Theta}^{(a)}}$ is the conditional expectation with respect to the current model parameters.

As is common with the EM formulation, we introduce a hidden assignment variable $\mathbf{z}_{i,j}$, which is an indicator variable for when the audio sample set Y_i is assigned to the jth component of $\Theta^{(a)}$, i.e., when $z_i^{(a)} = j$. The complete-data log-likelihood is then

$$\begin{aligned}
&\log p\left(X, Y, Z|\Theta^{(a)}\right) \\
&\qquad = \sum_{i=1}^{K^{(s)}} \sum_{j=1}^{K^{(a)}} \mathbf{z}_{i,j} \log \pi_j^{(a)} + \mathbf{z}_{i,j} \log p\left(Y_i, X_i|\Theta_j^{(a)}\right). \quad (10)
\end{aligned}$$

The $\mathcal{Q}$ function is then obtained by taking the conditional expectation of (10), and using the law of large numbers to remove the dependency on the virtual samples. The result is a $\mathcal{Q}$ function that depends only on the parameters of the song-level DTs $\Theta_i^{(s)}$. For the detailed derivation of HEM for DTM, we refer the reader to our earlier work [30], [32].

Algorithm 1 HEM algorithm for DTM

1: **Input**: combined song-level DTM $\{\pi_i^{(s)}, \Theta_i^{(s)}\}_{i=1}^{K^{(s)}}$, number of virtual samples N.
2: Initialize tag-level DTM $\{\hat{\pi}_j^{(a)}, \hat{\Theta}_j^{(a)}\}_{j=1}^{K^{(a)}}$.
3: **repeat**
4: {E-step}

5: Compute expectations using sensitivity analysis for each $\Theta_i^{(s)}$ and $\hat{\Theta}_j^{(a)}$ (see Appendix A and [30]):

$$\begin{aligned}
\hat{x}_{t|j}^{(i)} &= \mathbb{E}_{y|\Theta_i^{(s)}}\left[\mathbb{E}_{x|y,\hat{\Theta}_j^{(a)}}[x_t]\right] \\
\hat{P}_{t,t|j}^{(i)} &= \mathbb{E}_{y|\Theta_i^{(s)}}\left[\mathbb{E}_{x|y,\hat{\Theta}_j^{(a)}}\left[x_t x_t^T\right]\right] \\
\hat{P}_{t,t-1|j}^{(i)} &= \mathbb{E}_{y|\Theta_i^{(s)}}\left[\mathbb{E}_{x|y,\hat{\Theta}_j^{(a)}}\left[x_t x_{t-1}^T\right]\right] \\
\hat{W}_{t|j}^{(i)} &= \mathbb{E}_{y|\Theta_i^{(s)}}\left[\left(y_t - \hat{\hat{y}}_j^{(a)}\right)\mathbb{E}_{x|y,\hat{\Theta}_j^{(a)}}[x_t]^T\right] \\
\hat{U}_{t|j}^{(i)} &= \mathbb{E}_{y|\Theta_i^{(s)}}\left[\left(y_t - \hat{\hat{y}}_j^{(a)}\right)\left(y_t - \hat{\hat{y}}_j^{(a)}\right)^T\right] \\
\hat{u}_t^{(i)} &= \mathbb{E}_{y|\Theta_i^{(s)}}[y_t] \\
\ell_{i|j} &= \mathbb{E}_{\Theta_i^{(s)}}\left[\log p\left(y_{1:\tau}|\hat{\Theta}_j^{(a)}\right)\right].
\end{aligned} \tag{11}$$

6: Compute assignment probability and weighting:

$$\hat{\mathbf{z}}_{i,j} = \frac{\hat{\pi}_j^{(a)} \exp(N_i \ell_{i|j})}{\sum_{j'=1}^{K^{(a)}} \hat{\pi}_{j'}^{(a)} \exp(N_i \ell_{i|j'})} \tag{12}$$

$$\hat{\mathbf{w}}_{i,j} = \hat{\mathbf{z}}_{i,j} N_i = \hat{\mathbf{z}}_{i,j} \pi_i^{(s)} N. \tag{13}$$

7: Compute aggregate expectations for each $\hat{\Theta}_j^{(a)}$:

$$\begin{aligned}
\hat{N}_j &= \sum_i \hat{\mathbf{z}}_{i,j}, \quad \eta_j = \sum_i \hat{\mathbf{w}}_{i,j} \hat{P}_{1,1|j}^{(i)} \\
\hat{M}_j &= \sum_i \hat{\mathbf{w}}_{i,j}, \quad \gamma_j = \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=1}^{\tau} \hat{u}_t^{(i)} \\
\xi_j &= \sum_i \hat{\mathbf{w}}_{i,j} \hat{x}_{1|j}^{(i)}, \quad \beta_j = \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=1}^{\tau} \hat{x}_{t|j}^{(i)} \\
\Phi_j &= \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=1}^{\tau} \hat{P}_{t,t|j}^{(i)}, \\
\Psi_j &= \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=2}^{\tau} \hat{P}_{t,t-1|j}^{(i)} \\
\varphi_j &= \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=2}^{\tau} \hat{P}_{t,t|j}^{(i)} \\
\phi_j &= \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=2}^{\tau} \hat{P}_{t-1,t-1|j}^{(i)} \\
\Lambda_j &= \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=1}^{\tau} \hat{U}_{t|j}^{(i)} \\
\Gamma_j &= \sum_i \hat{\mathbf{w}}_{i,j} \sum_{t=1}^{\tau} \hat{W}_{t|j}^{(i)}.
\end{aligned} \tag{14}$$

8: {M-step}

9: Recompute parameters for each component $\hat{\Theta}_j^{(a)}$:

$$\begin{aligned}
\hat{C}_j^{(a)} &= \Gamma_j \Phi_j^{-1}, \hat{R}_j^{(a)} = \frac{1}{\tau \hat{M}_j}\left(\Lambda_j - \hat{C}_j^{(a)} \Gamma_j\right) \\
\hat{A}_j^{(a)} &= \Psi_j \phi_j^{-1}, \hat{Q}_j^{(a)} = \frac{1}{(\tau-1)\hat{M}_j}\left(\varphi_j - \hat{A}_j^{(a)} \Psi_j^T\right) \\
\hat{\mu}_j^{(a)} &= \frac{1}{\hat{M}_j}\xi_j, \hat{S}_j^{(a)} = \frac{1}{\hat{M}_j}\eta_j - \hat{\mu}_j^{(a)}\left(\hat{\mu}_j^{(a)}\right)^T \\
\hat{\pi}_j^{(a)} &= \frac{\hat{N}_j}{K^{(s)}}, \hat{\hat{y}}_j^{(a)} = \frac{1}{\tau \hat{M}_j}\left(\gamma_j - \hat{C}_j^{(a)} \beta_j\right).
\end{aligned} \tag{15}$$

10: **until** convergence

11: **Output**: tag-level DTM $\{\hat{\pi}_j^{(a)}, \hat{\Theta}_j^{(a)}\}_{j=1}^{K^{(a)}}$.

The HEM algorithm for DTM is summarized in Algorithm 1. In the E-step, the expectations in (11) are computed for each song-level component $\Theta_i^{(s)}$ and current tag-level component $\hat{\Theta}_j^{(a)}$. These expectations can be computed using "suboptimal filter analysis" or "sensitivity analysis" [33] on the Kalman smoothing filter (see Appendix A and [30]). Next, the probability of assigning the song-level component $\Theta_i^{(s)}$ to the tag-level component $\Theta_j^{(a)}$ is computed according to (12), and the expectations are aggregated over all the song-level DTs in (14). In the M-step, the parameters for each tag-level component $\hat{\Theta}_j^{(a)}$ are recomputed according to the update equations in (15). Note that the E- and M-steps for HEM-DTM are related to the standard EM algorithm for DTM. In particular, the song-level DT components $\Theta_i^{(s)}$ take the role of the "data-points" in standard EM. This is manifested in the E-step of HEM as the expectation with respect to $\Theta_i^{(s)}$, which averages over the possible values of the "data-points." Given the aggregate expectations, the parameter updates in the M-step of HEM and EM are identical.

VI. Music Datasets

In this section, we describe the music collection and the audio features used in our experiments.

A. CAL500 Database

The CAL500 [2] dataset consists of 502 popular Western songs from the last 50 years from 502 different artists. Each song has been annotated by at least three humans, using a semantic vocabulary of 149 tags that describe genres, instruments, vocal characteristics, emotions, acoustic characteristics, and song usages. CAL500 provides hard binary annotations, which are 1 when a tag applies to the song and 0 when the tag does not apply. We find empirically that accurately fitting the HEM-DTM model requires a significant number of training examples, due to the large number of parameters in the model. Hence, we restrict our attention to the 78 tags with at least 50 positively associated songs.

B. Swat10k Database

Swat10k [34] is a collection of over ten thousand songs from 4597 different artists, weakly labeled from a vocabulary of 18 genre tags, 135 sub-genre tags, and 475 other acoustic tags. The song-tag associations are mined from Pandora's website. Each song is labeled with 2 to 25 tags. As for CAL500, we restrict our attention to the tags (125 genre tags and 326 acoustic tags) with at least 50 positively associated songs.

C. Audio Features

Mel-frequency cepstral coefficients (MFCCs) [35] are a popular feature for content-based music analysis, which concisely summarize the short-time content of an acoustic waveform by using the discrete cosine transform (DCT) to decorrelate the bins

of a Mel-frequency spectrum.[2] In Section III-A, we noted how the DT model can be viewed as a time-varying PCA representation of the audio feature vectors. This suggests that we can represent the Mel-frequency spectrum over time as the output of the DT model y_t. In this case, the columns of the observation matrix C (a learned PCA matrix) are analogous to the DCT basis functions, and the hidden states x_t are the coefficients (analogous to the MFCCs). The advantage of learning the PCA representation, rather than using the standard DCT basis, is that different basis functions (C matrices) can be learned to best represent the particular song or semantic tag of interest. Hence, the DT can focus on the frequency structure that is relevant for modeling the particular tag. Another advantage of learning the basis functions is that it may allow a much smaller sized state transition matrix: using the DCT basis functions instead of the learned ones may require more basis functions to capture the timbral information and hence a higher-dimensional state vector. Estimating a smaller-sized state transition matrix is more efficient and expected to be less prone to overfitting. The benefits of learning the basis functions will be validated in Section VII-C (see, e.g., Table VII). Also, note that since the DT explicitly models the temporal evolution of the audio features, we do not need to include their instantaneous derivatives (as in the MFCC deltas).

In our experiments, we use 34 Mel-frequency bins, computed from half-overlapping, 46-ms windows of audio. The Mel-frequency bins are represented in a dB scale, which accurately accounts for the human auditory response to acoustic stimuli. Each audio fragment is described by a time series $y_{1:\tau}^t$ of $\tau = 450$ sequential audio feature vectors, which corresponds to 10 s. Song-level DTM models are learned from a dense sampling of audio fragments of 10 s, extracted every 1 second.

VII. Experimental Evaluation

In this section, we present results on music annotation and retrieval using the DTM model.

A. Experimental Setup

We set the state-space dimension $n = 7$, as in the work by Barrington *et al.* [11]. Song-level DTMs are learned with $K^{(s)} = 16$ components to capture enough of the temporal diversity present in each song, using EM-DTM [12]. Tag-level DTMs are learned by pooling together all song-level models associated with a given tag and reducing the result to a DTM with $K^{(a)} = 2$ components with HEM-DTM. We keep $K^{(a)}$ low to prevent HEM-DTM from overfitting (compared to HEM-GMM, HEM-DTM requires estimating significantly more parameters per mixture component). Section VII-C illustrates that the system is fairly robust for reasonable variations in these parameters.

The EM-DTM algorithm to estimate song-level DTMs follows an iterative "component splitting" procedure. First, a one-component mixture is estimated by initializing parameters randomly and running EM until convergence. Then, the number of components is increased by splitting this component and EM is run to convergence again. This process of splitting components and re-running EM for a mixture with more components is repeated until the desired number of components is obtained. When splitting a component, new components are initialized by replicating the component and slightly perturbing—randomly and differently for each new component—the poles of the state transition matrix A. We follow a growing schedule of $\{1, 2, 4, 8, 16\}$ mixture components. The single component of the initial mixture is learned from a set of randomly selected fragments of the song, using the method proposed by Doretto *et al.* [9]. This "component splitting" procedure for EM-DTM was found to be quite robust to different initializations. More details can be found in earlier work by Chan *et al.* [12]. The tag-level DTMs (with $K^{(a)} = 2$ components) are learned by running ten trials of the HEM-DTM algorithm. Each trial is initialized by randomly selecting two mixture components from the aggregated song-level mixtures. The final parameter estimates are obtained from the trial that achieves the highest likelihood. This procedure proved robust as well.

To investigate the advantage of the DTM's temporal representation, we compare the auto-tagging performance of HEM-DTM to the hierarchically trained Gaussian mixture models (HEM-GMMs) [2], the CBA model [4], the boosting approach [6], and the SVM approach [5]. We follow the original procedure for training HEM-GMM and CBA, with the modification that the CBA codebook is constructed using only songs from the training set. We report performance also for direct-EM model estimation (EM-DTM), which learns each tag-level DTM model using the standard EM algorithm for DTM [12] directly on a subsampled set of all audio fragments associated with the tag. Empirically, we found that due to RAM requirements a single run of EM-DTM only manages to process about 1% of the data (i.e., audio fragments) that HEM-DTM can process, when estimating a tag model from approximately 200 training examples, on a modern laptop with 4 GB of RAM. In contrast, HEM-DTM, through the estimation of intermediate models, can pool over a much richer training data set, both in the number of songs and in the density of audio fragments sampled within each song. Finally, we compare to model aggregation DTM (AGG-DTM), which estimates each tag-level model by aggregating all the song-level DTM models associated with the tag. A drawback of this technique is that the number of DTs in the tag-level DTM models grows linearly with the size of the training set, resulting in drawn out delays in the evaluation stage. All reported metrics are the results of five-fold cross validation where each song appeared in the test set exactly once.

B. Evaluation of Annotation and Retrieval

Annotation performance is measured following the procedure described by Turnbull *et al.* [2]. Test set songs are annotated with the ten most likely tags in their semantic multinomial (4). Annotation accuracy is reported by computing precision, recall and F-score for each tag,[3] and then averaging over all tags. Per-tag precision is the probability that the model correctly uses the tag when annotating a song. Per-tag recall is the probability that the

[2]This decorrelation is usually convenient in that it reduces the number of parameters to be estimated.

[3]We compute annotation metrics on a *per-tag* basis, as our goal is to build an automatic tagging algorithm with high stability over a wide range of semantic tags. *Per-song* metrics may get artificially inflated by consistently annotating songs with a small set of highly frequent tags, while ignoring less common tags.

model annotates a song that should have been annotated with the tag. Precision, recall and F-score measure for a tag w are defined as

$$\mathrm{P} = \frac{|w_C|}{|w_A|}, \mathrm{R} = \frac{|w_C|}{|w_H|}, \mathrm{F} = 2((\mathrm{P})^{-1} + (\mathrm{R})^{-1})^{-1} \qquad (16)$$

where $|w_H|$ is the number of tracks that have w in the ground truth, $|w_A|$ is the number of times our annotation system uses w when automatically tagging a song, and $|w_C|$ is the number of times w is correctly used. In case a tag is never selected for annotation, the corresponding precision (that otherwise would be undefined) is set to the tag prior from the training set, which equals the performance of a random classifier.

To evaluate retrieval performance, we rank-order test songs for each single-tag query in our vocabulary, as described in Section IV. We report mean average precision (MAP), area under the receiver operating characteristic curve (AROC) and top-10 precision (P10), averaged over all the query tags. The ROC curve is a plot of true positive rate versus false positive rate as we move down the ranked list. The AROC is obtained by integrating the ROC curve, and it is upper bounded by 1. Random guessing would result in an AROC of 0.5. The top-10 precision is the fraction true positives in the top-10 of the ranking. MAP averages the precision at each point in the ranking where a song is correctly retrieved.

C. Results on CAL500

Annotation and retrieval results on the CAL500 data set are presented in Table I. For all metrics, except for precision, the best performance is observed with HEM-DTM. For retrieval, while some other methods show a comparable AROC score, HEM-DTM clearly improves the top of the ranked list compared to any other method. The higher precision-at-10 score demonstrates this. The results also show that sub-sampling of the training set for direct-EM estimation (EM-DTM) degrades the performance, compared to HEM estimation (HEM-DTM). Aggregating the song-level DTMs associated with a tag (AGG-DTM) is also inferior. Fig. 3 plots the precision-recall curves, for annotation, for all methods. At low recall (shorter, more selective annotations), (H)EM-DTM outperforms any other method. At higher recall, HEM-GMM catches up. In future work, we will investigate whether combining DTM- and GMM-based annotations can make for a more accurate auto-tagger.

To illustrate how different values of $K^{(s)}$ and $K^{(a)}$ (the number of components in the song and tag mixture models, respectively) affect the system's performance, we vary $K^{(s)}$ in $\{2,4,8,16\}$ while fixing $K^{(a)} = 2$, and, vice versa, vary $K^{(a)}$ in $\{2,4,8,16\}$ while fixing $K^{(s)} = 16$. Annotation and retrieval results are reported in Table II, showing that performance is fairly robust within a reasonable parameter range.

We expect DTMs to be particularly beneficial for tags with characteristic temporal dynamics (e.g., tempo, rhythm, etc.) that unfold in the time span of a few seconds. Tags that are modeled adequately already by instantaneous spectral characteristics within a window of 50 ms (e.g., timbre) may not benefit much, as well as tags that might require a global, structured song model.

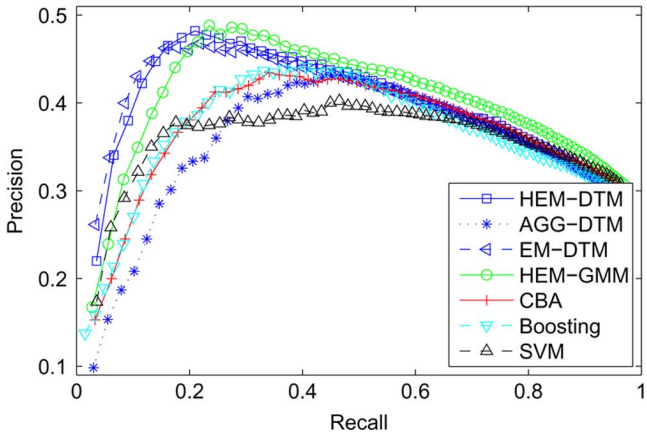

Fig. 3. Precision-recall curves for different methods. (H)EM-DTM dominates at low recall. GMMs catch up at higher recall.

TABLE I
ANNOTATION AND RETRIEVAL RESULTS FOR VARIOUS ALGORITHMS ON THE CAL500 DATA SET

Model	Annotation			Retrieval		
	P	R	F-score	AROC	MAP	P10
HEM-GMM [2]	**0.49**	0.23	0.26	0.66	0.45	0.47
CBA [4]	0.41	0.24	0.25	**0.69**	0.47	0.49
Boosting [6]	0.37	0.17	0.20	**0.69**	0.47	0.49
SVM [5]	0.38	0.24	0.25	0.66	0.45	0.48
HEM-DTM	0.47	**0.25**	**0.30**	**0.69**	**0.48**	**0.53**
AGG-DTM	0.34	0.23	0.21	**0.69**	0.47	0.50
EM -DTM	0.46	0.24	0.28	0.65	0.44	0.49

TABLE II
ANNOTATION AND RETRIEVAL PERFORMANCE AS A FUNCTION OF $K^{(s)}$ AND $K^{(a)}$, RESPECTIVELY

(a)				
$K^{(s)}$	2	4	8	16
MAP	0.4748	0.4768	0.4804	0.4834
F-score	0.2785	0.2886	0.2963	0.3002
(b)				
$K^{(a)}$	2	4	8	16
MAP	0.4834	0.4796	0.4820	0.4798
F-score	0.3002	0.2940	0.2942	0.2872

To illustrate this point, Table III lists annotation (F-score) and retrieval (MAP) results for a subset of the CAL500 vocabulary. DTMs prove suitable for tags with significant temporal structure, e.g., vocal characteristics and instruments such as electric or acoustic guitar, by capturing the attack/sustain/decay/release profile of the instruments. DTMs also capture the temporal characteristics of a "fast" song—expected to unfold "fast," i.e., within a few seconds—and significantly improve upon GMMs, which cannot model these characteristics. For "slow" songs, on the other hand, DTMs are not picking up any additional information that GMMs do not capture already. The same is observed when predicting tags such as "light beat" and "mellow," already well described by timbre information (as evidenced by the high GMM performance), or "weak" and "sad," where neither DTMs

TABLE III
ANNOTATION AND RETRIEVAL RESULTS FOR SOME TAGS WITH HEM-DTM AND HEM-GMM

Tag	HEM-DTM		HEM-GMM [2]	
	F-score	MAP	F-score	MAP
HEM-DTM outperforms HEM-GMM				
female lead vocals	0.58	0.69	0.42	0.44
male lead vocals	0.44	0.87	0.08	0.81
backing vocals	0.30	0.47	0.09	0.44
emotional vocals	0.37	0.42	0.05	0.27
vocal harmonies	0.21	0.21	0.10	0.15
pop	0.32	0.36	0.31	0.35
acoustic guitar	0.44	0.44	0.31	0.43
electric guitar	0.32	0.37	0.14	0.33
fast	0.40	0.48	0.20	0.42
positive feelings	0.30	0.51	0.10	0.48
driving	0.29	0.38	0.29	0.33
HEM-DTM under-performs HEM-GMM				
light beat	0.36	0.58	0.53	0.61
mellow	0.34	0.41	0.37	0.49
slow	0.45	0.60	0.44	0.62
weak	0.22	0.26	0.26	0.25
sad	0.13	0.23	0.28	0.30
negative feelings	0.27	0.29	0.32	0.30
going to sleep	0.27	0.33	0.35	0.36

nor GMMs are capturing strongly predictive acoustic characteristics. While, for some tags, this may indicate that "timbre tells all," for others, capturing more specific characteristics might require modeling structure at a much longer time scale or higher level. This will be a topic of future research.

It should also be noted that the increased modeling power of DTMs, compared to GMMs, requires more training data to reliably estimate them. This is discussed in more detail later in this section. Especially when training data is more noisy (e.g., for more subjective tags), significantly more examples will be required to make stand out the salient attributes HEM-DTM is trying to capture. This may explain why DTMs improve over GMMs for "positive feelings" (over 170 examples in CAL500) but not for "negative feelings (less than 80 examples). The same consideration holds for the usage tags "driving" and "going to sleep," which respectively appear 141 and 56 times in CAL500. So, while DTMs can capture a superset of the information modeled by GMMs, they may still perform worse for some tags, for this reason. Another factor to keep in mind when observing worse DTM than GMM performance is the more limited modeling power of DTMs when no clear temporal dynamics are present. Indeed, the absence of clear regularities in the temporal dynamics will result in a degenerate linear dynamical system (e.g., $A = 0$), reducing each DT component to a Gaussian component. Clearly, a DTM with two Gaussian components is a less rich timbre model than a GMM with 16 mixture components (as proposed by Turnbull *et al.* [2]). Estimating a DTM with as many mixture components, on the other hand, is prone to overfitting. This would result in a poorer timbre model as well.

TABLE IV
AUTOMATIC TEN-TAG ANNOTATIONS FOR DIFFERENT SONGS. CAL500 GROUND TRUTH ANNOTATIONS ARE MARKED IN BOLD

Kool and the Gang - "Funky stuff"	
HEM-DTM	**danceable**, **positive feelings**, driving, **party**, **cheerful**, **happy**, **energy**, **light**, **catchy**, **lighthearted**
HEM-GMM	weak, alternative, synthesized, **danceable**, major, cold, **indifferent**, driving, synthesizer, **unromantic**
Stevie Ray Vaughan - "Pride and joy"	
HEM-DTM	**drum set**, **romantic**, emotional vocals, **recommended**, **positive**, **like**, **cheerful**, emotional, **light**, alternative
HEM-GMM	weak, alternative, synthesized, synthesizer, not happy, unromantic, negative feelings, major, indifferent, cold
The Beach Boys - "I get around"	
HEM-DTM	**synthesizer**, **synthesized**, party, **electronica**, danceable, **heavy beat**, arousing, rough, **pop**, fast
HEM-GMM	**synthesized**, **electronica**, **synthesizer**, **pop**, party, cold, happy, danceable, **heavy beat**, **uptight**
The Police - "Every little thing she does is magic"	
HEM-DTM	**classic rock**, **driving**, **energy**, **fast**, **male lead vocals**, **electric guitar**, **electric**, indifferent, powerful, rough
HEM-GMM	boring, major, acoustic, **driving**, not likeable, female lead vocals, **recording quality**, cold, **synthesized**, pop , guitar
R.E.M. - "Camera"	
HEM-DTM	romantic, piano, **touching**, tender, pop, **pleasant**, **positive**, backing vocals, light beat, **calming**
HEM-GMM	weak, alternative, major, piano, synthesized, synthesizer, pop, female lead vocals, backing vocals, not likeable

Table IV reports automatic ten-tag annotations for some songs from the CAL500 music collection, with HEM-DTM and HEM-GMM. Tables V and VI show the Top-10 retrieval results for the queries "acoustic guitar" and "female lead vocals," respectively, both for HEM-DTM and HEM-GMM. For "acoustic guitar," it is noted that both GMM and DTM make some "acceptable" mistakes. For example, "Golden brown," by The Stranglers, has a harpsichord, and Aaron Neville's "Tell it like it is" has clean electric guitar.

We investigate how the size of the training set affects the quality of the resulting models. As suggested earlier, reliably estimating the more powerful but also more complex DTM models is expected to require more training examples, compared to estimating GMM models. Fig. 4 illustrates this. In Fig. 4(a), we consider all 149 CAL500 tags and plot the relative retrieval performance of HEM-DTM, compared to HEM-GMM, for tag subsets of different minimal cardinality. The cardinality of a tag is

TABLE V
TOP-10 RETRIEVED SONGS FOR "ACOUSTIC GUITAR." SONGS WITH ACOUSTIC GUITAR ARE MARKED IN BOLD

Rank	HEM-DTM	
1	The Zombies	"Beechwood park"
2	**James Taylor**	**"Fire and rain"**
3	**Arlo Guthrie**	**"Alice's restaurant massacree"**
4	**Crosby, Stills, Nash & Young**	**"Teach your children"**
5	**American Music Club**	**"Jesus hands"**
6	**Donovan**	**"Catch the wind"**
7	Smokey Robinson & The Miracles	"Ooo baby baby"
8	Aaron Neville	"Tell it like it is"
9	The Animals	"I'm crying"
10	**10cc**	**"For you and I"**
Rank	HEM-GMM	
1	The Stranglers	"Golden brown"
2	**Crosby, Stills, Nash & Young**	**"Teach your children"**
3	Pet Shop Boys	"Being boring"
4	The Police	"Every little thing she does is magic"
5	Belle and Sebastian	"Like Dylan in the movies"
6	Counting Crows	"Speedway"
7	The Beautiful South	"One last love song"
8	Beth Quist	"Survival"
9	Neutral Milk Hotel	"Where you'll find me now"
10	**James Taylor**	**"Fire and rain"**

TABLE VI
TOP-10 RETRIEVED SONGS FOR "FEMALE LEAD VOCALS." SONGS WITH FEMALE LEAD VOCALS ARE MARKED IN BOLD

Rank	HEM-DTM	
1	**Artemis**	**"Don't look back"**
2	**The Ronettes**	**"Walking in the rain"**
3	**Alicia Keys**	**"Fallin"**
4	**Syreeta**	**"What love has joined together"**
5	**Dionne Warwick**	**"Walk on by"**
6	**Gloria Gaynor**	**"I will survive"**
7	**Anita Baker**	**"Caught up in the rapture"**
8	**The Andrews Sisters**	**"Boogie woogie bugle boy"**
9	**The Buggles**	**"Video killed the radio star"**
10	Antonio Carlos Jobim	"Wave"
Rank	HEM-GMM	
1	**Artemis**	**"Don't look back"**
2	**Britney Spears**	**"I'm a slave for you"**
3	**Alicia keys**	**"Fallin"**
4	**Dionne Warwick**	**"Walk on by"**
5	**Syreeta**	**"What love has joined together"**
6	The Beach Boys	"I get around"
7	Kourosh Zolani	"Peaceful planet"
8	**Natalie Imbruglia**	**"Torn"**
9	**Cat Power**	**"He war"**
10	The Animals	"I'm crying"

defined as the number of examples in the data set that are associated with the tag. The minimal cardinality of a set of tags is determined by its tag with lowest cardinality. The plot shows that DTM modeling provides a bigger performance boost, over GMMs, when more examples are available for a tag. This is confirmed in Fig. 4(b). This experiment is restricted to the ten CAL500 tags that have cardinality of 150 or more. For each tag, the size of the training set is varied from 25 to 150, by random subsampling. Finally, the average retrieval performance (over these ten tags) is reported as a function of the training set size, both for HEM-DTM and HEM-GMM. Initially, a larger training set benefits both methods. However, while GMM performance levels off beyond 100 training examples, DTM performance keeps improving. Additional training examples keep leveraging more of the DTM's extra modeling potential, widening the gap between DTM and GMM performance.

Finally, we validate our claim that learning the observation matrix C (i.e., the basis functions for the Mel-spectrum), rather than using the standard DCT basis, is beneficial as it combines a better representation of the features of the Mel-spectrum with a more compact model of the temporal dynamics that are characteristic for a particular song or semantic tag of interest. In Table VII, we compare HEM-DTM, with a learned C-matrix, with HEM-DTM-DCT, where we modify the DT model to fix the observation matrix C to be the DCT basis functions. We report annotation and retrieval performance for an experimental setup similar to the one in Table I, with $n = 7$, $K^{(s)} = 16$, and $K^{(a)} = 2$. For HEM-DTM-DCT, the first $n = 7$ DCT bases (ordered by frequency) are selected. We also analyze the effect of a higher-dimensional DCT basis for HEM-DTM-DCT, by increasing n to 13. HEM-DTM outperforms both HEM-DTM-DCT variants, which illustrates that learning the observation matrix C improves the performance over using a standard DCT basis. The small difference in performance between HEM-DTM-DCT for $n = 7$ and $n = 13$, respectively, suggests that overfitting on the (higher-dimensional) hidden state process may be neutralizing the benefits of a larger (fixed) basis, which allows to better represent the Mel-frequency spectrum, for $n = 13$.

D. Results on Swat10k

HEM-DTM scales well to larger music collections, like this data set. It efficiently estimates tag models from a large number of examples by breaking the problem down into intermediate steps. The annotation and retrieval results on Swat10k, presented in Table VIII, demonstrate that this procedure to estimate DTMs is also accurate. On Swat10k, DTMs outperform GMMs for every performance metric reported,[4] except for precision on the "acoustic" tags. The annotation results are obtained by annotating songs with the two most likely "genre" tags (ideally one main genre and one sub-genre), and with the ten most likely acoustic tags. Precision-recall curves are shown in Fig. 5, confirming the overall dominance of HEM-DTM over HEM-GMM for the annotation task. In summary, for both Swat10k tag categories, DTMs successfully capture temporal

[4] Swat10k is weakly labeled, i.e., song annotations are incomplete. Given enough positive training examples, this does not affect the estimation of *generative* models (see, e.g., [2], [24]), like GMM and DTM. For evaluation purposes, while this still allows relative comparisons, it will reduce the absolute value of some performance metrics, e.g., MAP and P10 that evaluate positive song-tag associations at the top of the ranking.

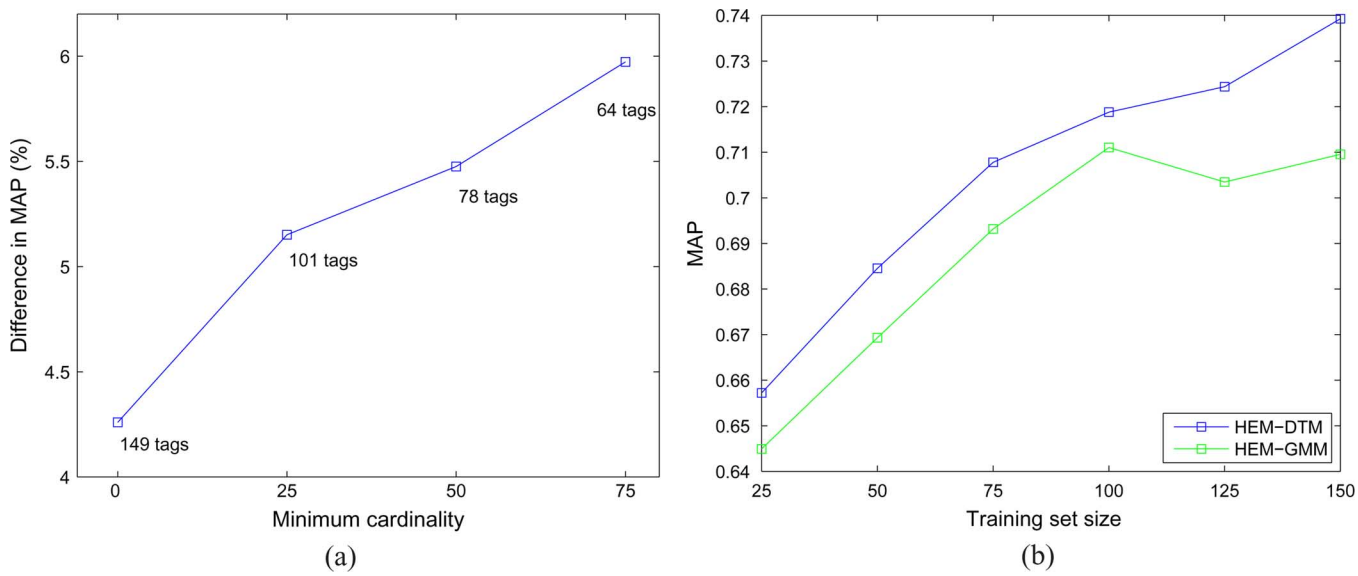

Fig. 4. (a) Retrieval performance of HEM-DTM, relative to HEM-GMM, as a function of the minimal cardinality of tag subsets. More precisely, for each point in the graph, the set of all 149 CAL500 tags is restricted to those that CAL500 associates with a number of songs that is at least the abscissa value. The number of tags in each restricted subset is indicated next to the corresponding point in the graph. (b) Retrieval performance, averaged over the 10 CAL500 tags that have cardinality of 150 or more, as a function of training set size. Training sets of size $25, 50, \ldots, 150$ are randomly subsampled.

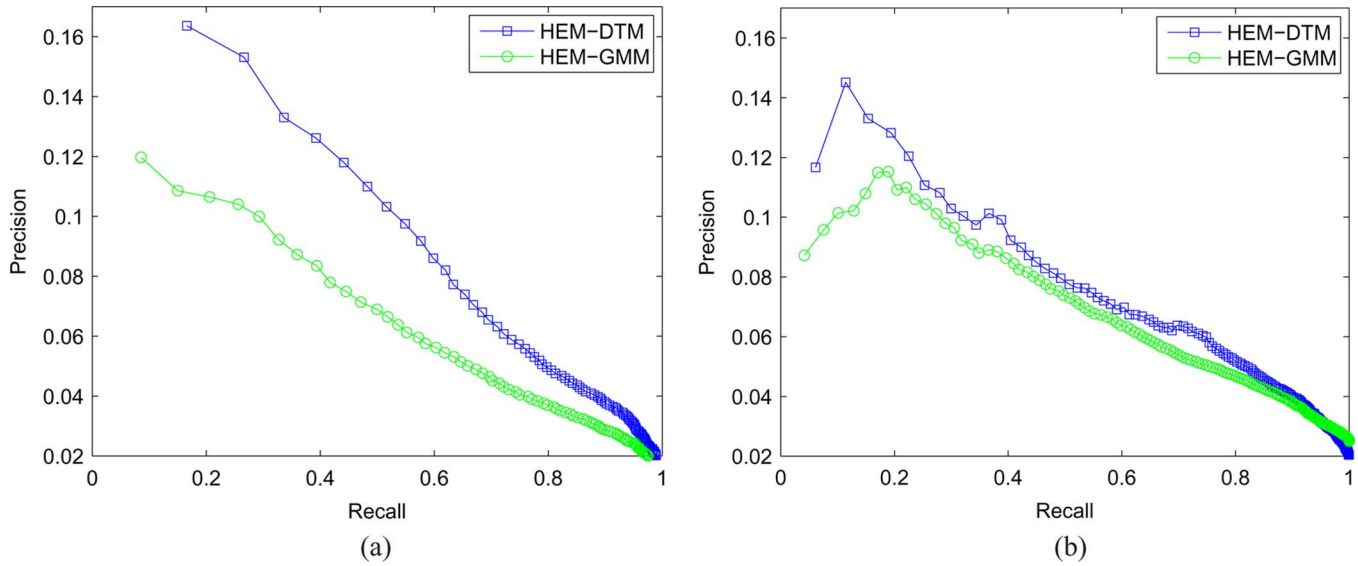

Fig. 5. Precision-recall curves for annotation experiments on Swat10k, for both tag categories. (a) "Genre" tags. (b) "Acoustic" tags.

TABLE VII
ANNOTATION AND RETRIEVAL RESULTS FOR HEM-DTM AND HEM-DTM-DCT ($n = 7$ AND $n = 13$)

Model	Annotation			Retrieval		
	P	R	F-score	AROC	MAP	P10
HEM-DTM	**0.47**	**0.25**	**0.30**	**0.69**	**0.48**	**0.53**
HEM-DTM-DCT (n=7)	0.43	0.21	0.21	0.67	0.45	0.47
HEM-DTM-DCT (n=13)	0.42	0.21	0.21	0.68	0.45	0.47

TABLE VIII
ANNOTATION AND RETRIEVAL RESULTS ON THE SWAT10K DATA SET, FOR BOTH TAG CATEGORIES

Model	P	R	F-score	AROC	MAP	P10
"Genre"						
HEM-DTM	**0.15**	**0.27**	**0.16**	**0.87**	**0.14**	**0.18**
HEM-GMM	0.11	0.15	0.08	0.82	0.11	0.15
"Acoustic"						
HEM-DTM	0.10	**0.34**	**0.13**	**0.82**	**0.13**	**0.17**
HEM-GMM	**0.11**	0.24	0.10	0.77	0.10	0.14

dynamics over a few seconds as well as instantaneous timbre information, providing more accurate models.

VIII. DISCUSSION ON THE DTM MODEL'S PARAMETERS

In this section, we illustrate qualitatively how variations in the acoustic characteristics of semantic tags are reflected in different DTM model parameters. We show how the dynamics of a musical tag affect the state transition matrix A and how the structure of the observation matrix C specializes for different

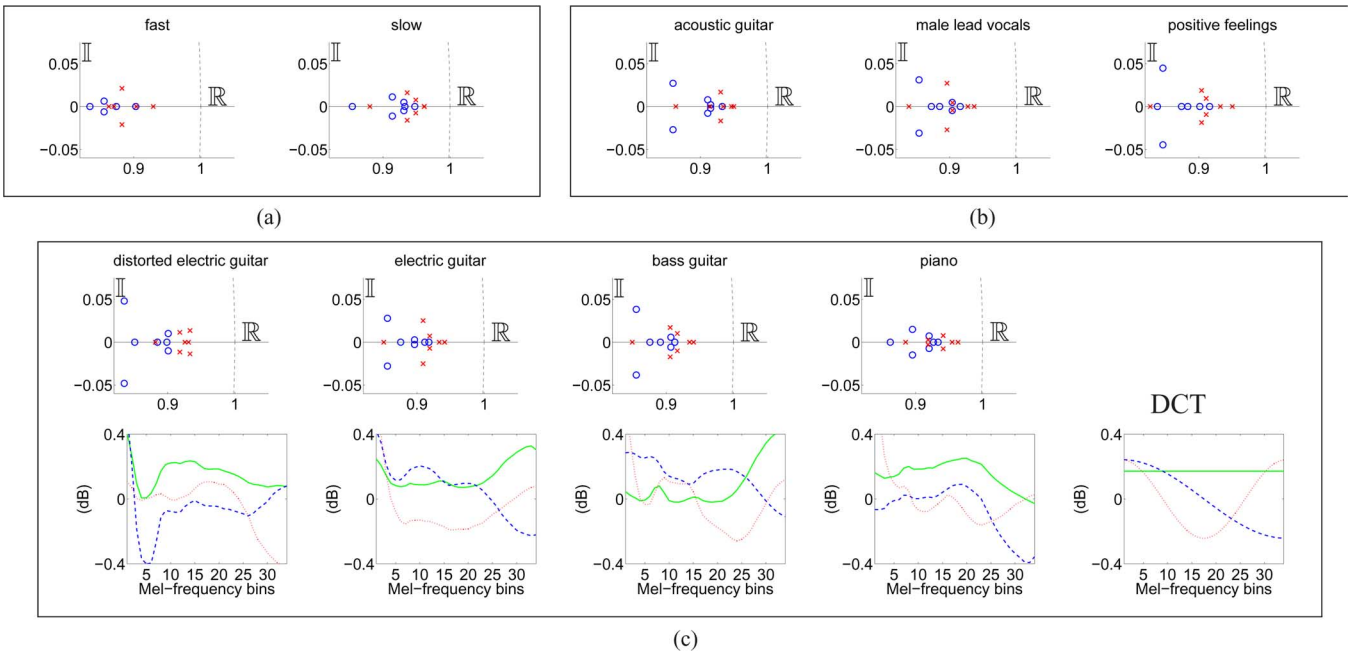

Fig. 6. Location of the poles of the DTM models for different tags (blue circles and red crosses correspond to different DT components of the DTM). The horizontal and vertical axes of the figures represent the real and imaginary parts of the poles, respectively. The angle between each of the conjugate poles and the positive real axis determines the normalized frequency. (a) "Fast" shows higher normalized frequencies than "Slow." (b) HEM-DTM captures clear dynamics for tags in the upper portion of Table III, by modeling distinct normalized frequencies. (c) (Top row) Similar instruments are modeled with similar normalized frequencies. (Bottom row) Timbral characteristics are modeled by the observation matrix C. The first three columns of C are depicted in solid green, dashed blue, and dotted red line, for the corresponding tags in the top row. The columns of C define a basis that is optimized to best represent the instantaneous audio content for each tag. For comparison, the standard DCT basis (used to compute MFCCs) is shown on the far right. (a) "Fast" versus "Slow." (b) Poles for some other tag models. (c) Different types of guitar, and piano.

tags. We also present some two-dimensional embeddings of tag models, showing that qualitatively similar musical tags give rise to qualitatively similar DTM models.

A. *State Transition Matrix: Temporal Dynamics*

Doretto *et al.* [36] describe the link between the location of the poles[5] of the state transition matrix, A, and the dynamics of the LDS. The higher a normalized frequency (i.e., the wider the angle between each of the conjugate poles and the positive real axis), the faster and more distinct the associated dynamics. On the other hand, if all the poles are on the positive real axis, there are no dynamics connected with the modes of the system. Second, the distances of the poles from the origin control the durations of the corresponding modes of the system. Poles closer to the origin require stronger excitement for their mode to persist in the system.

Fig. 6(a) sketches the poles of the mixture components of the DTM models for the tags "Fast" and "Slow" respectively, from the experiment of Section VII. The location of the poles in the polar plane is in agreement with the intuition that the former is characterized by faster dynamics while the latter coincides with smoother variations. Fig. 6(b) shows the location of the poles for some of the tags in the upper portion of Table III, for which HEM-DTM shows improvements over HEM-GMM.

[5]Consider the decomposition $A = P\Lambda P^{-1}$, where Λ is a diagonal matrix containing the eigenvalues of A, and P is an orthogonal matrix whose columns are the corresponding eigenvectors. The eigenvalues are the poles of the system. The eigenvectors determine the corresponding modes, describing the system's characteristic oscillations.

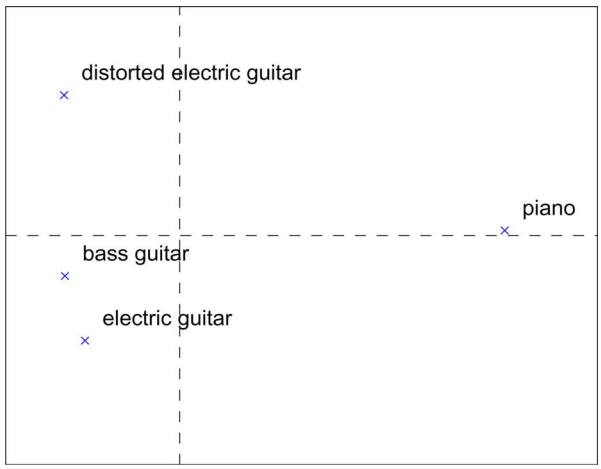

Fig. 7. Each point represents the audio feature subspace of a different tag. The axes are unlabeled since only the relative positions of the points are relevant. The relative positions reflect similarity between the subspaces based on Martin distance. Guitar-type models are more similar, and clearly separated from the piano model.

HEM-DTM associates some distinct normalized frequencies with these tags. Finally, Fig. 6(c) (top row) plots the poles for different types of guitar (electric, distorted electric and bass) and piano. The figure illustrates that acoustically similar instruments have similar dynamics. For example, the locations of the poles of acoustic guitar and clean electric guitar are fairly similar. Also, the various types of guitar show similar dynamics, while "Piano" is characterized by noticeably different normalized frequencies.

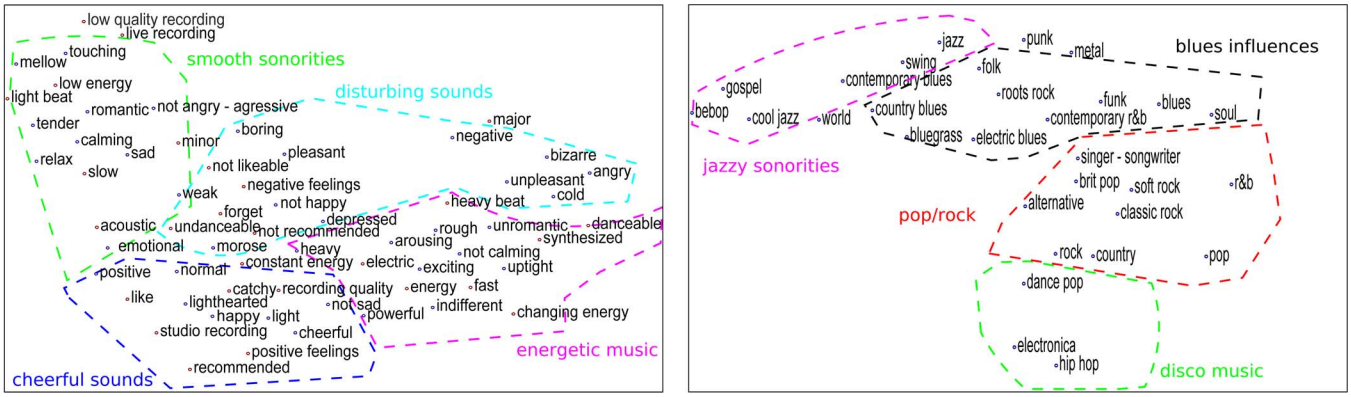

Fig. 8. Two-dimensional embeddings of DTM-based tag models based on t-SNE and symmetrized KL divergence. Relative positions are qualitatively consistent with the semantic meaning of the tags. (a) "Emotion" and "Acoustic characteristics" tags. The top-left tip hosts smooth acoustic sensations. In the bottom prevails cheerful music and, moving right, energetic sounds. (b) "Genre" tags. The center gathers pop-rock sonorities. Moving towards the top-left tip, this evolves to sophisticated jazzy sounds. Hip hop, electronica and dance music are at the bottom of the plot.

B. Observation Matrix: Instantaneous Spectral Content

While the state transition matrix A encodes rhythm and tempo, the observation matrix C accounts for instantaneous timbre. In particular, a DT model generates features into the subspace[6] spanned by the columns of the observation matrix.

The bottom row of Fig. 6(c) displays the first three basis vectors for the guitar tags and the piano tag in the top row. Each semantic tag is modeled by distinct basis functions that fit its particular music qualities and timbre. In contrast, the DCT basis functions used for MFCCs are fixed *a priori*. When modeling Mel-frequency spectra as the output of a DTM model, dimensionality reduction and model estimation are coupled together. On the other hand, for MFCCs, the DCT is performed before model estimation and is not adapted to specific audio sequences.

C. DTM: Timbre and Dynamics

For a more intuitive interpretation, Fig. 7 represents the audio feature subspaces—whose first three basis functions are depicted in the bottom row of Fig. 6(c)—for the previous guitar and piano tags as a point in a two-dimensional embedding. The relative positions of the points reflect the Martin distance ([37], [38]) between the DTs for the corresponding tags, which is related to the difference in principal angles of the observation matrices [39].

The figure indicates that DTM tag models corresponding to qualitatively similar tags generate audio features in similar subspaces. For example, note that the guitar-type models are well separated from the piano model along the horizontal axis, while smaller variations along the vertical coordinate are observed between the different types of guitars.

Tag-level DTMs (combining state transitions with an observation model) simultaneously model both the timbre and dynamics of a tag. In this subsection, we qualitatively examine how similar/different the resulting models are for different tags. In particular, t-SNE [40] is used to embed tag models in a two-dimensional space, based on the Kullback–Leibler (KL) divergence between DTMs. The KL divergence between two DTs can be computed efficiently with a recursive formula [41]. The KL divergence between two mixture models, not analytically tractable in exact form, can be approximated efficiently (see, e.g., [42]).

Fig. 8 shows two-dimensional embeddings for different groups of tags, learned from CAL500. Fig. 8(a) displays the embedding for "emotion" and "acoustic characteristics" tags. Qualitatively, the resulting embedding is consistent with the semantic meaning of the different tags. For example, the top-left protuberance of the cloud gathers tags associated with smooth acoustic sensations: from low-energy and romancing sounds, to relaxing, slow sonorities. In the bottom of the cloud prevails a sense of happiness, that, marching to the right of the plot, turns into energy, excitement, and heavy beats.

Similarly, Fig. 8(b) provides the two-dimensional embedding for tags of the category "genre." In the center of the cloud, there is a strong rock and pop influence, usually characterized by the sound of electric guitars, the beat of drums, and melodies sung by male lead vocalists. Moving toward the top-left of the graph, music fills up with emotions typical of blues and finally evolves to more sophisticated jazzy sounds. In the bottom, we find hip hop, electronica and dance music, more synthetic sonorities often diffused in night clubs.

IX. Conclusion

In this paper, we have proposed a novel approach to automatic music annotation and retrieval that captures temporal (e.g., rhythmical) aspects as well as timbral content. In particular, our approach uses the dynamic texture mixture model, a generative time series model for musical content, as a tag-level annotation model. To learn the tag-level DTMs, we use a two-step procedure: 1) learn song-level DTMs from individual songs using the EM algorithm (EM-DTM); 2) learn the tag-level DTMs using a hierarchical EM algorithm (HEM-DTM) to summarize the common information shared by song-level DTMs associated with a tag. This hierarchical learning procedure is efficient and easily parallelizable, allowing DTM tag models to be learned from large sets of weakly labeled songs (e.g., up to 2200 songs per tag in our experiments).

[6]This is exactly true when the observation noise is ignored or negligible, i.e., $R \to 0$.

Experimental results demonstrate that the new DTM tag model improves accuracy over current bag-of-features approaches (e.g., GMMs, shown on the first line of Table I), which do not model the temporal dynamics in a musical signal. In particular, we see significant improvements in tags with temporal structures that span several seconds, e.g., vocal characteristics, instrumentation, and genre. This leads to more accurate annotation at low recall, and improvements in retrieval at the top of the ranked-list. While, in theory, DTMs are a more general model than GMMs (as a DTM with degenerate temporal dynamics is equivalent to a GMM), we observe that in some cases GMMs are favorable. For musical characteristics that do not have distinctive long-term temporal dynamics, a GMM with more mixture components may be better suited to model pure timbre information, since it *a priori* ignores longer-term dynamics. A DTM with the same number of components, on the other hand, may overfit to the temporal dynamics of the training data, resulting in a poorer timbre model. Preventing overfitting by using a DTM with less mixture components *a priori* limits its flexibility as a pure timbre model though. This suggests that further gains are possible by using both DTM (longer-term temporal) and GMM (short-term timbre) annotation models. Future work will address this topic by developing criteria for selecting a suitable annotation model for a specific tag, or by combining results from multiple annotation models using the probabilistic formalism inherent in the generative models. Finally, our experiments show that DTM tag models perform significantly better when more training data is available. As an alternative to supplying more data, future work will consider learning DTM tag models using a Bayesian approach, e.g., by specifying suitable (data-driven) prior distributions on the DTM parameters, thus reducing the amount of training data required to accurately model the musical tag.

APPENDIX
EXPECTATIONS FOR THE E-STEP OF HEM-DTM

The expectations in (11) for each combination of $\Theta_i^{(s)}$ and $\Theta_j^{(a)}$ can be computed efficiently using sensitivity analysis on the Kalman smoothing filter. Let $\Theta_s = \{A_s, C_s, Q_s, R_s, \mu_s, S_s, \overline{y}_s\}$ and $\Theta_a = \{A_a, C_a, Q_a, R_a, \mu_a, S_a, \overline{y}_a\}$ be the DT parameters for components $\Theta_i^{(s)}$ and $\Theta_j^{(a)}$, respectively. The procedure is outlined in Algorithm 2 (derivations appear in [30], [32]). First, we use the Kalman smoothing filter (Algorithm 3) to compute the conditional covariances

$$\begin{aligned}
\hat{V}_{t|\tau}^{(s)} &= \mathrm{cov}_{x|y,\Theta_s}(x_t) \\
\hat{V}_{t,t-1|\tau}^{(s)} &= \mathrm{cov}_{x|y,\Theta_s}(x_t, x_{t-1}) \\
\hat{V}_{t|\tau}^{(a)} &= \mathrm{cov}_{x|y,\Theta_a}(x_t) \\
\hat{V}_{t,t-1|\tau}^{(a)} &= \mathrm{cov}_{x|y,\Theta_a}(x_t, x_{t-1})
\end{aligned} \tag{17}$$

and intermediate filter parameters $\{F_t^{(s)}, G_t^{(s)}, H_t^{(s)}\}$, $\{F_t^{(a)}, G_t^{(a)}, H_t^{(a)}\}$, for both Θ_s and Θ_a. The notation $\hat{x}_{t|s}^{(a)}$ denotes the expectation at time t, conditioned on the sequence $y_{1:s}$, with respect to Θ_a. Next, sensitivity analysis of the Kalman filter (Algorithm 4) computes the mean and variance of the one-step ahead state estimators when $y_{1:t-1} \sim \Theta_s$

$$\hat{\mathbf{x}}_t = \mathbb{E}_{\Theta_s}\begin{bmatrix} x_t^{(s)} \\ \hat{x}_{t|t-1}^{(s)} \\ \hat{x}_{t|t-1}^{(a)} \end{bmatrix}, \hat{\mathbf{V}}_t = \mathrm{cov}_{\Theta_s}\left(\begin{bmatrix} x_t^{(s)} \\ \hat{x}_{t|t-1}^{(s)} \\ \hat{x}_{t|t-1}^{(a)} \end{bmatrix}\right). \tag{18}$$

The notation $\mathbf{V}^{[i,j]}$ refers to the (i,j) matrix in the block matrix $\mathbf{V}$, and $\mathbf{x}^{[i]}$ refers to the ith vector in the block vector $\mathbf{x}$. Next, sensitivity analysis of the Kalman smoothing filter (Algorithm 5) computes the mean and variance of the state estimators for the full sequence $y_{1:\tau} \sim \Theta_s$

$$\begin{aligned}
\hat{x}_t^{(s)} &= \mathbb{E}_{y|\Theta_s}\left[\hat{x}_{t|\tau}^{(a)}\right], \quad \hat{\kappa}_t^{(s)} = \mathrm{cov}_{y|\Theta_s}\left(y_t, \hat{x}_{t|\tau}^{(a)}\right) \\
\hat{V}_t^{(s)} &= \mathbb{E}_{y|\Theta_s}\left[\hat{V}_{t|\tau}^{(a)}\right], \quad \hat{V}_{t,t-1}^{(s)} = \mathbb{E}_{y|\Theta_s}\left[\hat{V}_{t,t-1|\tau}^{(a)}\right] \\
\hat{\chi}_t^{(s)} &= \mathrm{cov}_{y|\Theta_s}\left(\hat{x}_{t|\tau}^{(a)}\right), \quad \hat{\chi}_{t,t-1}^{(s)} = \mathrm{cov}_{y|\Theta_s}\left(\hat{x}_{t|\tau}^{(a)}, \hat{x}_{t-1|\tau}^{(a)}\right).
\end{aligned} \tag{19}$$

Finally, given the quantities in (18), (19), the E-step expectations and expected log-likelihood are computed according to (20) and (21).

Algorithm 2 Expectations for HEM-DTM

1: **Input**: DT parameters Θ_s and Θ_a, length τ.
2: Run Kalman smoothing filter (Algorithm 3) on Θ_s and on Θ_a.
3: Run sensitivity analysis on Θ_s and Θ_a for the Kalman filter and Kalman smoothing filter (Algorithms 4 and 5).
4: Compute E-step expectations, for $t = \{1, \ldots, \tau\}$:

$$\begin{aligned}
\hat{u}_t^{(s)} &= C_s \hat{\mathbf{x}}^{[1]}{}_t + \overline{y}_s \\
\hat{U}_t^{(s)} &= C_s \hat{\mathbf{V}}_t^{[1,1]} C_s^T + R_s + \left(\hat{u}_t^{(s)} - \overline{y}_a\right)\left(\hat{u}_t^{(s)} - \overline{y}_a\right)^T \\
\hat{P}_t^{(s)} &= \hat{V}_{t|\tau}^{(a)} + \hat{\chi}_t^{(s)} + \hat{x}_t^{(s)}\left(\hat{x}_t^{(s)}\right)^T \\
\hat{P}_{t,t-1}^{(s)} &= \hat{V}_{t,t-1|\tau}^{(a)} + \hat{\chi}_{t,t-1}^{(s)} + \hat{x}_t^{(s)}\left(\hat{x}_{t-1}^{(s)}\right)^T \\
\hat{W}_t^{(s)} &= \hat{\kappa}_t^{(s)} + \left(\hat{u}_t^{(s)} - \overline{y}_a\right)\left(\hat{x}_t^{(s)}\right)^T.
\end{aligned} \tag{20}$$

5: Compute expected log-likelihood ℓ:

$$\begin{aligned}
\hat{\Lambda}_t &= C_a\left(\hat{\mathbf{V}}_t^{[3,3]} + \hat{\mathbf{x}}_t^{[3]}\left(\hat{\mathbf{x}}_t^{[3]}\right)^T\right)C_a^T \\
\hat{\Sigma}_t &= C_a \hat{V}_{t|t-1}^{(a)} C_a^T + R_a \\
\hat{\lambda}_t &= C_s \hat{\mathbf{V}}_t^{[2,3]} C_a^T + \left(C_s \hat{\mathbf{x}}_t^{[1]} + \overline{y}_s - \overline{y}_a\right)\left(\hat{\mathbf{x}}_t^{[3]}\right)^T C_a^T \\
\ell &= \sum_{t=1}^{\tau} -\frac{1}{2}\,\mathrm{tr}\left(\hat{\Sigma}_t^{-1}\left(\hat{U}_t^{(s)} - \hat{\lambda}_t - \hat{\lambda}_t^T + \hat{\Lambda}_t\right)\right) \\
&\quad - \frac{1}{2}\log|\hat{\Sigma}_t| - \frac{m}{2}\log(2\pi).
\end{aligned} \tag{21}$$

6: **Output**: $\{\hat{x}_t^{(s)}, \hat{P}_t^{(s)}, \hat{P}_{t,t-1}^{(s)}, \hat{W}_t^{(s)}, \hat{U}_t^{(s)}, \hat{u}_t^{(s)}\}, \ell$.

Algorithm 3 Kalman smoothing filter

1: **Input**: DT parameters $\Theta = \{A, C, Q, R, \mu, S, \bar{y}\}$, length τ.
2: Initialize: $\hat{V}_{1|0} = S$.
3: **for** $t = \{1, \ldots, \tau\}$ **do**
4: {Kalman filter—forward recursion}

$$\begin{aligned}\hat{V}_{t|t-1} &= A\hat{V}_{t-1|t-1}A^T + Q\\ K_t &= \hat{V}_{t|t-1}C^T(C\hat{V}_{t|t-1}C^T + R)^{-1}\\ \hat{V}_{t|t} &= (I - K_tC)\hat{V}_{t|t-1}\\ G_t &= AK_t, \quad F_t = A - AK_tC.\end{aligned}$$

5: **end for**
6: Initialize: $\hat{V}_{\tau,\tau-1|\tau} = (I - K_\tau C)A\hat{V}_{\tau-1|\tau-1}$.
7: **for** $t = \{\tau, \ldots, 2\}$ **do**
8: {Kalman smoothing filter—backward recursion}

$$\begin{aligned}J_{t-1} &= \hat{V}_{t-1|t-1}A^T(\hat{V}_{t|t-1})^{-1}\\ H_{t-1} &= A^{-1} - J_{t-1}\\ \hat{V}_{t-1|\tau} &= \hat{V}_{t-1|t-1} + J_{t-1}(\hat{V}_{t|\tau} - \hat{V}_{t|t-1})J_{t-1}^T\\ \hat{V}_{t-1,t-2|\tau} &= \hat{V}_{t-1|t-1}J_{t-2}^T\\ &\quad + J_{t-1}(\hat{V}_{t,t-1|\tau} - A\hat{V}_{t-1|t-1})J_{t-2}^T.\end{aligned}$$

9: **end for**
10: **Output**: $\{\hat{V}_{t|t-1}, \hat{V}_{t|t}, \hat{V}_{t|\tau}, \hat{V}_{t,t-1|\tau}, G_t, F_t, H_t\}$.

Algorithm 4 Sensitivity analysis of the Kalman filter

1: **Input**: DTs Θ_s and Θ_a, associated Kalman filters, length τ.
2: Initialize: $\hat{\mathbf{x}}_1 = \begin{bmatrix}\mu_s\\ \mu_s\\ \mu_a\end{bmatrix}$, $\hat{\mathbf{V}}_1 = \begin{bmatrix}S_s & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}$.
3: **for** $t = \{2, \ldots, \tau + 1\}$
4: Form block matrices:

$$\mathbf{A}_{t-1} = \begin{bmatrix}A_s & 0 & 0\\ G_{t-1}^{(s)}C_s & F_{t-1}^{(s)} & 0\\ G_{t-1}^{(a)}C_s & 0 & F_{t-1}^{(a)}\end{bmatrix}, \mathbf{B} = \begin{bmatrix}I\\ 0\\ 0\end{bmatrix}$$

$$\mathbf{C}_{t-1} = \begin{bmatrix}0\\ G_{t-1}^{(s)}\\ G_{t-1}^{(a)}\end{bmatrix}, \mathbf{D}_{t-1} = \begin{bmatrix}0\\ 0\\ G_{t-1}^{(a)}\end{bmatrix}.$$

5: Update means and covariances:

$$\begin{aligned}\hat{\mathbf{x}}_t &= \mathbf{A}_{t-1}\hat{\mathbf{x}}_{t-1} + \mathbf{D}_{t-1}(\bar{y}_s - \bar{y}_a),\\ \hat{\mathbf{V}}_t &= \mathbf{A}_{t-1}\hat{\mathbf{V}}_{t-1}\mathbf{A}_{t-1}^T + \mathbf{B}Q_s\mathbf{B}^T + \mathbf{C}_{t-1}R_s\mathbf{C}_{t-1}^T.\end{aligned}$$

6: **end for**
7: **Output**: $\{\hat{\mathbf{x}}_t, \hat{\mathbf{V}}_t\}$.

Algorithm 5 Sensitivity analysis of the Kalman smoothing filter

1: **Input**: DTs Θ_s and Θ_a, associated Kalman smoothing filter, and Kalman filter sensitivity analysis, length τ.
2: Initialize: $\hat{x}_\tau^{(s)} = A_a^{-1}\hat{\mathbf{x}}_{\tau+1}^{[3]}$, $\hat{\chi}_\tau^{(s)} = A_a^{-1}\hat{\mathbf{V}}_{\tau+1}^{[3,3]}A_a^{-T}$, $L_\tau = A_a^{-1}$, $M_\tau = \mathbf{0}$.
3: **for**$t = \{\tau, \ldots, 1\}$ **do**
4: Compute cross-covariance:

$$\rho_t = \left(L_tF_t^{(a)}\hat{\mathbf{V}}_t{}^{[3,2]} + (L_tG_t^{(a)}C_s + M_t)\hat{\mathbf{V}}_t^{[1,1]}\right)C_s^T + L_tG_t^{(a)}R_s.$$

5: **if** $t > 1$ **then**
6: Compute sensitivity:

$$\begin{aligned}\omega_t &= L_tF_t^{(a)}\hat{\mathbf{V}}_t^{[3,3]} + \left(L_tG_t^{(a)}C_s + M_t\right)\hat{\mathbf{V}}_t^{[2,3]}\\ \hat{x}_{t-1}^{(s)} &= H_{t-1}^{(a)}\hat{\mathbf{x}}^{[3]}{}_t + J_{t-1}^{(a)}\hat{x}_t^{(s)}\\ \hat{\chi}_{t-1}^{(s)} &= \begin{bmatrix}H_{t-1}^{(a)} & J_{t-1}^{(a)}\end{bmatrix}\begin{bmatrix}\hat{\mathbf{V}}_t^{[3,3]} & \omega_t^T\\ \omega_t & \hat{\chi}_t^{(s)}\end{bmatrix}\begin{bmatrix}\left(H_{t-1}^{(a)}\right)^T\\ \left(J_{t-1}^{(a)}\right)^T\end{bmatrix}\\ \hat{\chi}_{t,t-1}^{(s)} &= \omega_t\left(H_{t-1}^{(a)}\right)^T + \hat{\chi}_t^{(s)}\left(J_{t-1}^{(a)}\right)^T.\end{aligned}$$

7: Update matrices:

$$\begin{aligned}L_{t-1} &= H_{t-1}^{(a)} + J_{t-1}^{(a)}L_tF_t^{(a)}\\ M_{t-1} &= J_{t-1}^{(a)}\left(L_tG_t^{(a)}C_s + M_t\right)A_s.\end{aligned}$$

8: **end if**.
9: **end for**
10: **Output**: $\{\hat{x}_t^{(s)}, \hat{\chi}_t^{(s)}, \hat{\chi}_{t,t-1}^{(s)}, \hat{\kappa}_t^{(s)} = \rho_t^T\}$.

ACKNOWLEDGMENT

The authors thank the editor and reviewers for their constructive comments, M. Mandel for assistance with the implementation of the SVM auto-tagging algorithm [5], T. Bertin-Mahieux and M. Hoffman for providing code for the boosting [6] and CBA [4] algorithms respectively, and L. Barrington, V. Mahadevan, Brian. McFee, and M. Slaney for helpful discussions.

REFERENCES

[1] M. Goto and K. Hirata, "Recent studies on music information processing," *Acoust. Sci. Technol.*, vol. 25, no. 6, pp. 419–425, 2004.
[2] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 467–476, Feb. 2008.
[3] S. Ness, A. Theocharis, G. Tzanetakis, and L. Martins, "Improving automatic music tag annotation using stacked generalization of probabilistic SVM outputs," in *Proc. ACM Multimedia*, 2009.
[4] M. Hoffman, D. Blei, and P. Cook, "Easy as CBA: A simple probabilistic model for tagging music," in *Proc. ISMIR*, 2009, pp. 369–374.
[5] M. Mandel and D. Ellis, "Multiple-instance learning for music information retrieval," in *Proc. ISMIR*, 2008, pp. 577–582.
[6] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green, "Automatic generation of social tags for music recommendation," in *Adv. Neural Inf. Process. Syst.*, 2007.
[7] M. Casey, C. Rhodes, and M. Slaney, "Analysis of minimum distances in high-dimensional musical spaces," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 5, pp. 1015–1028, 2008.
[8] M. McKinney and J. Breebaart, "Features for audio and music classification," in *Proc. ISMIR*, 2003, pp. 154–158.
[9] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *Intl. J. Comput. Vision*, vol. 51, no. 2, pp. 91–109, 2003.
[10] J. Reed and C. Lee, "A study on music genre classification based on universal acoustic models," in *Proc. ISMIR*, 2006, pp. 89–94.
[11] L. Barrington, A. Chan, and G. Lanckriet, "Modeling music as a dynamic texture," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 18, no. 3, pp. 602–612, Mar. 2010.
[12] A. B. Chan and N. Vasconcelos, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 909–926, May 2008.
[13] S. Essid, G. Richard, and B. David, "Inferring efficient hierarchical taxonomies for MIR tasks: Application to musical instruments," in *Proc. ISMIR*, 2005, pp. 324–328.

[14] B. Whitman and R. Rifkin, "Musical query-by-description as a multiclass learning problem," in *Proc. IEEE Multimedia Signal Process. Conf.*, Dec. 2002, pp. 153–156.
[15] P. Cano and M. Koppenberger, "Automatic sound annotation," in *Proc. IEEE Signal Process. Soc. Workshop Mach. Learn. Signal Process.*, 2004, pp. 391–400.
[16] M. Slaney, "Mixtures of probability experts for audio retrieval and indexing," in *Proc. IEEE Multimedia and Expo.*, 2002, pp. 345–348.
[17] L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet, "Combining feature kernels for semantic music retrieval," in *Proc. ISMIR*, 2008.
[18] E. Pampalk, A. Flexer, and G. Widmer, "Improvements of audio-based music similarity and genre classification," in *Proc. ISMIR*, 2005, pp. 628–633.
[19] A. Flexer, F. Gouyon, S. Dixon, and G. Widmer, "Probabilistic combination of features for music classification," in *Proc. ISMIR*, 2006, pp. 111–114.
[20] M. Slaney, K. Weinberger, and W. White, "Learning a metric for music similarity," in *Proc. ISMIR*, 2008, pp. 313–318.
[21] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 5, pp. 293–302, Jul. 2002.
[22] B. Whitman and D. Ellis, "Automatic record reviews," in *Proc. ISMIR*, 2004, pp. 470–477.
[23] A. Berenzweig, B. Logan, P. W. Ellis, D. Whitman, and B. Whitman, "A large-scale evaluation of acoustic and subjective music-similarity measures," *Comput. Music J.*, vol. 28, no. 2, pp. 63–76, 2004.
[24] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos, "Supervised learning of semantic classes for image annotation and retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 3, pp. 394–410, Mar. 2007.
[25] J. Aucouturier and F. Pachet, "Music similarity measures: What's the use?," in *Proc. ISMIR*, 2002, pp. 157–163.
[26] M. Hoffman, D. Blei, and P. Cook, "Content-based musical similarity computation using the hierarchical Dirichlet process," in *Proc. ISMIR*, 2008, pp. 349–354.
[27] R. H. Shumway and D. S. Stoffer, "An approach to time series smoothing and forecasting using the EM algorithm," *J. Time Series Anal.*, vol. 3, no. 4, pp. 253–264, 1982.
[28] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Towards musical query-by-semantic description using the cal500 data set," in *Proc. ACM SIGIR*, 2007.
[29] N. Vasconcelos and A. Lippman, "Learning mixture hierarchies," in *Proc. Adv. Neural Inf. Process. Syst.*, 1998.
[30] A. Chan, E. Coviello, and G. Lanckriet, "Clustering dynamic textures with the hierarchical EM algorithm," in *Proc. IEEE CVPR*, 2010, pp. 2022–2029.
[31] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Statist. Soc. B*, vol. 39, pp. 1–38, 1977.
[32] A. B. Chan, E. Coviello, and G. Lanckriet, "Derivation of the hierarchical EM algorithm for dynamic textures," City Univ. of Hong Kong, Tech. Rep., 2010.
[33] A. Gelb, *Applied Optimal Estimation*. Cambridge, MA: MIT Press, 1974.
[34] D. Tingle, Y. E. Kim, and D. Turnbull, "Exploring automatic music annotation with "acoustically objective" tags," in *Proc. MIR*, New York, 2010, pp. 55–62, ACM.
[35] B. Logan, "Mel frequency cepstral coefficients for music modeling," in *Proc. ISMIR*, 2000, vol. 28.
[36] G. Doretto and S. Soatto, "Editable dynamic textures," in *Proc. IEEE CVRP*, Jun. 2003, vol. 2, pp. 137–142.
[37] P. Saisan, G. Doretto, Y. N. Wu, and S. Soatto, "Dynamic texture recognition," in *Proc. Los Alamitos*, CA, 2001, pp. 58–65.
[38] K. D. Cock, K. D. Cock, B. D. Moor, and B. D. Moor, "Subspace angles between linear stochastic models," in *Proc. IEEE CDC*, 2000, pp. 1561–1561.
[39] L. Wolf and A. Shashua, "Learning over sets using kernel principal angles," *J. Mach. Learn. Res.*, vol. 4, pp. 913–931, 2003.
[40] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.
[41] A. B. Chan and N. Vasconcelos, "Probabilistic kernels for the classification of auto-regressive visual processes," in *Proc. IEEE CVRP*, Washington, DC, 2005, pp. 846–851, IEEE Computer Society.
[42] J. Hershey and P. Olsen, "Approximating the Kullback-Leibler divergence between Gaussian mixture models," in *IEEE ICASSP 2007*, 2007, pp. 317–320.

Emanuele Coviello received the "Laurea Triennale" degree in information engineering and the "Laurea Specialistica" degree in telecommunication engineering from the Universitá degli Studi di Padova, Padova, Italy, in 2006 and 2008, respectively. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering, University of California at San Diego (UCSD), La Jolla, where he has joined the .

He is currently with the Computer Audition Laboratory, Department of Electrical and Computer Engineering, UCSD.

Mr. Coviello received the "Premio Guglielmo Marconi Junior 2009" award, from the Guglielmo Marconi Foundation (Italy), and won the "2010 Yahoo! Key Scientific Challenge Program," sponsored by Yahoo!. His main interest is machine learning applied to music information retrieval and multimedia data modeling.

Antoni B. Chan (M'08) received the B.S. and M.Eng. degrees in electrical engineering from Cornell University, Ithaca, NY, in 2000 and 2001, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California, San Diego (UCSD), La Jolla, in 2008.

From 2001 to 2003, he was a Visiting Scientist in the Vision and Image Analysis Lab, Cornell University, and in 2009, he was a Postdoctoral Researcher in the Statistical Visual Computing Lab at UCSD. In 2009, he joined the Department of Computer Science at the City University of Hong Kong, as an Assistant Professor.

Dr. Chan was the recipient of an NSF IGERT Fellowship from 2006 to 2008. His research interests are in computer vision, machine learning, pattern recognition, and music analysis.

Gert Lanckriet received the M.S. degree in electrical engineering from the Katholieke Universiteit Leuven, Leuven, Belgium, in 2000 and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 2001 and 2005, respectively.

In 2005, he joined the Department of Electrical and Computer Engineering, University of California, San Diego, where he heads the Computer Audition Laboratory. His research focuses on the interplay of convex optimization, machine learning, and signal processing, with applications in computer audition and music information retrieval.

Prof. Lanckriet was awarded the SIAM Optimization Prize in 2008 and is the recipient of a Hellman Fellowship and an IBM Faculty Award.