Modeling Music as a Dynamic Texture

Luke Barrington, *Student Member, IEEE*, Antoni B. Chan, *Member, IEEE*, and Gert Lanckriet

***Abstract*—We consider representing a short temporal fragment of musical audio as a *dynamic texture*, a model of both the timbral and rhythmical qualities of sound, two of the important aspects required for automatic music analysis. The dynamic texture model treats a *sequence* of audio feature vectors as a sample from a linear dynamical system. We apply this new representation to the task of automatic song segmentation. In particular, we cluster audio fragments, extracted from a song, as samples from a dynamic texture mixture (DTM) model. We show that the DTM model can both accurately cluster coherent segments in music and detect transition boundaries. Moreover, the generative character of the proposed model of music makes it amenable for a wide range of applications besides segmentation. As examples, we use DTM models of songs to suggest possible improvements in other music information retrieval applications such as music annotation and similarity.**

***Index Terms*—Automatic segmentation, dynamic texture model (DTM), music modeling, music similarity.**

I. Introduction

MODELS of music begin with a representation of the audio content in some machine-readable form. It is common practice in music information retrieval to represent a song as an unordered set or "bag" of audio feature vectors (e.g., Mel-frequency cepstral coefficients). While this has shown promise in many applications, (e.g., music annotation and retrieval [1], audio similarity [2] and song segmentation [3]), the bag-of-feature-vectors representation is fundamentally limited by ignoring the time-dependency between feature vectors. Permuting the feature vectors in the bag will not alter the representation, so information encapsulated in how feature vectors are ordered in time is ignored. As a result, the bag-of-feature-vectors representation fails to represent the higher level, longer term musical dynamics of an audio fragment, like rhythmic qualities (e.g., tempo and beat patterns) and temporal structure (e.g., repeated riffs and arpeggios).

In this paper, we address the limitations of the bag-of-features representation by modeling simultaneously the instantaneous spectral content (timbre) as well as the longer term spectral dynamics (rhythmic and temporal structure) of audio fragments that are several seconds in length [4]. To do this, we propose to use a *dynamic texture* (DT) [5] to represent a *sequence* of audio feature vectors as a sample from a generative probabilistic model, specifically, a linear dynamical system (LDS).

One application where it is useful to model the temporal, as well as timbral, dynamics of music is automatic song segmentation; the task of dividing a song into self-coherent units which a human listener would label as similar (e.g., verse, chorus, bridge, etc.). In particular, we propose a new algorithm that segments a song by clustering fragments of the song's audio content, using a *dynamic texture mixture* (DTM) model [6]. We test the segmentation algorithm on a wide variety of songs from two popular music datasets, and show that the dynamic texture captures much of the information required to determine the structure of music.

We also illustrate the applicability of the DTM segmentation to other music information retrieval problems. For example, one common problem with semantic song annotation (auto-tagging) occurs when different segments of the same song contain a variety of musical styles and instrumentations (the "Bohemian Rhapsody problem"). For such songs, the bag-of-features representation averages musical information from the whole song and existing auto-tagging systems (e.g., [1]) will produce generic descriptions of the song. One solution to this problem is first to segment the song into its constituent parts using the proposed automatic segmentation algorithm, and then to generate tags for each segment. We show that the dynamic texture model produces musical segments with homogeneous timbre and tempo, resulting in a more precise description of the song.

The remainder of this paper is organized as follows. In Section II, we review related work on song segmentation. In Section III, we introduce the dynamic texture models for audio fragments, and in Section IV we propose an algorithm for segmenting song structure using the DTM. Section V evaluates the segmentation algorithm on two music datasets. Finally, Section VI illustrates several applications of song segmentation to music annotation, retrieval, and visualization.

II. Related Work

The goal of automatic song segmentation is to divide a song into self-coherent units such as the chorus, verse, bridge, etc. Foote [7] segments music based on self-similarity between timbre features. Paulus and Klapuri [8] efficiently search the space of all possible segmentations and use a musicological model to label the most plausible segmentation.

Other methods attempt to model music explicitly and then cast segmentation as a clustering problem. Gaussian mixture models (GMMs) ignore temporal relations between features but model music well for applications such as music segmentation and similarity [3] as well as classification of a variety of semantic musical attributes [1]. Hidden Markov models (HMMs) consider transitions between feature states and have offered improvements for segmentation [9], key phrase detection [10] and genre classification [11]. Abdallah *et al.* [12] incorporate

Manuscript received January 08, 2009; revised October 18, 2009. Current version published February 10, 2010. The work of L. Barrington and A. B. Chan was supported by the National Science Foundation (NSF) under Grant IGERT DGE-0333451. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Bertrand David.

The authors are with the Department of Electrical and Computer Engineering, University of California, San Diego, CA 92093 USA (e-mail: lukeinusa@gmail.com).

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.2009.2036306

prior knowledge about segment duration into a HMM clustering model to address the problem of over-segmentation. Levy and Sandler [13] realize that feature-level HMMs do not capture sufficient temporal information so encode musical segments as clusters of HMM state-sequences and improve their clustering using constraints based on the temporal length of musical segments.

The DT model used in this paper is similar to the HMM, in that they are both probabilistic time-series models with hidden states that evolve over time. The main difference between the two models is that the hidden states of the HMM take on *discrete values*, whereas those of the DT are *real-valued vectors*. As a consequence, the HMM representation discretizes the observations into bins defined by the observation likelihoods, and the evolution of the sequence is modeled as jumps between these bins. The *continuous* state space of the DT, on the other hand, can capture smooth (rather than discrete) dynamics of state transitions and model the observed audio fragments without quantization.

Structural segmentation of music is often used as a first step in discovering distinctive or repeated sections that can serve as a representative summary or musical thumbnail of both acoustic [10], [14], [15] and symbolic [16] music representations. For example, Bartsch and Wakefield [17] follow [7] but use chroma features to identify repeated segments for audio thumbnailing and Goto adds high-level assumptions about repeated sections to build a system for automatically detecting choruses [18].

Similar to song segmentation is the task of detecting boundaries between musical segments (e.g., the change from verse to chorus). Turnbull *et al.* [19] present both an unsupervised (picking peaks of difference features) and supervised (boosted decision stumps) method for identifying musical segment boundaries. Similarly, Ong and Herrera [20] look for novelty in successive feature vectors to predict segment boundaries. These methods only detect the segment boundaries and make no attempt to assess the similarity of resulting segments.

Our formulation of treating audio as a dynamic texture was originally introduced in [4]. The current paper goes beyond [4] in the following ways: 1) we include a complete description of our segmentation algorithm; 2) we add a new step to the algorithm that uses music-based constraints to smooth the segments; 3) we present additional experiments on the PopMusic dataset from [13], along with illustrative examples; and 4) we include additional and more rigorous experiments on automatic annotation of music segments as opposed to entire songs.

III. Dynamic Textures Models

Consider representing the audio fragment in Fig. 1(a) with the corresponding sequence of audio feature vectors shown in Fig. 1(b). We would like to use these features to model simultaneously the instantaneous audio content (e.g., the instrumentation and timbre) and the melodic and rhythmic content (e.g., guitar riff, drum patterns, and tempo). In this paper, we will model the temporal dependencies in the audio *fragment* using a single model for the entire *sequence* of feature vectors. In particular, we will treat the sequence of feature vectors as a sample from a linear dynamical system (LDS). The LDS contains two random variables: 1) an observed variable, which generates the feature vector at each time-step (i.e., the instantaneous audio);

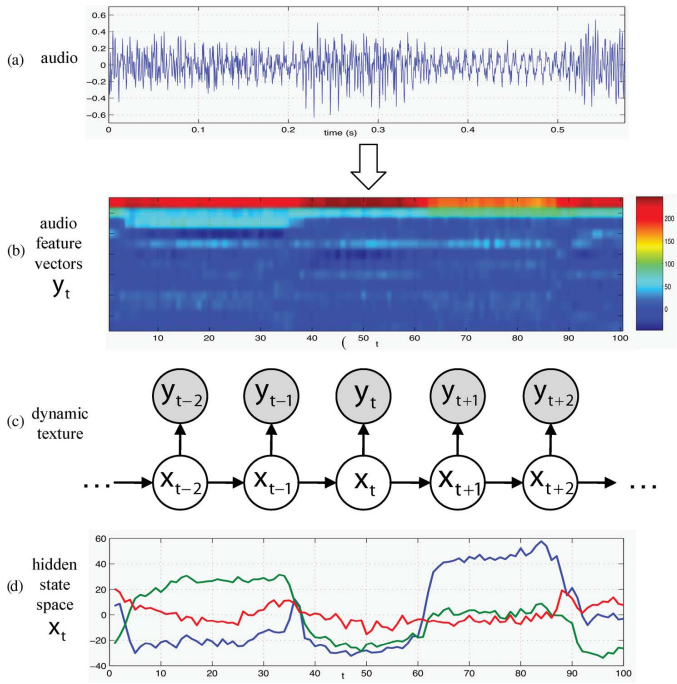

Fig. 1. Modeling audio as a temporal texture. (a) An audio waveform, and (b) feature vectors y_t extracted from the audio. (c) The sequence of features vectors $\{y_t\}$ is modeled as the output of a linear dynamical system, where (d) the hidden state-space sequence $\{x_t\}$ encodes both the instantaneous sound texture and the evolution of this texture over time.

and 2) a hidden variable which models the higher level musical state and how it dynamically evolves over time (i.e., the melodic and rhythmic content). In this way, we are able to capture *both the spectral and temporal properties* of the musical signal in a single probabilistic generative model.

The treatment of a time-series as a sample from a linear dynamical system is also known as a *dynamic texture* (DT) [5] in the computer vision literature, where a video is modeled as a sequence of vectorized image frames. The dynamic texture model has been successfully applied to various computer vision problems, including video texture synthesis [5], video recognition [21], [22], and motion segmentation [6], [23]. Although the DT was originally proposed in the computer vision literature as a generative model of video sequences, it is a generic model that can be applied to any time-series data, which in our case, are sequences of feature vectors that represent fragments of musical audio.

A. Dynamic Textures

A dynamic texture [5] is a generative model that treats a vector time-series as a sample from a linear dynamical system (LDS). Formally, the model captures both the appearance and the dynamics of the sequence with two random variables: an *observed variable* $y_t \in \mathbb{R}^m$, which encodes the appearance component (feature vector at time t); and a *hidden state variable* $x_t \in \mathbb{R}^n$ (with $n < m$), which encodes higher level characteristics of the time-series and their dynamics (sequence evolution over time). The state and observed variables are related through the *linear dynamical system* (LDS) defined by

$$\begin{cases} x_t = Ax_{t-1} + v_t \\ y_t = Cx_t + w_t + \bar{y} \end{cases} \quad (1)$$

where $A \in \mathbb{R}^{n\times n}$ is a *state transition matrix*, which encodes the dynamics of the hidden state, $C \in \mathbb{R}^{m\times n}$ is an *observation matrix*, which maps the hidden state variable to an observed feature vector, and $\bar{y} \in \mathbb{R}^m$ is the mean of the observed feature vectors, or the constant offset of the observation variable, y_t. The *driving noise process* v_t is normally distributed with zero mean and covariance Q, i.e., $v_t \sim \mathcal{N}(0, Q)$, where $Q \in \mathbb{S}_+^n$ is a positive definite $n \times n$ matrix, with $\mathbb{S}_+^n$ the set of positive definite matrices of dimension $n \times n$. The *observation noise* w_t is also zero mean and Gaussian, with covariance R, i.e., $w_t \sim \mathcal{N}(0, R)$, where $R \in \mathbb{S}_+^m$. The initial state vector x_1, which determines the starting point of the model, is distributed according to $x_1 \sim \mathcal{N}(\mu, S)$, with $\mu \in \mathbb{R}^n$ and $S \in \mathbb{S}_+^n$. The dynamic texture is specified by parameters $\Theta = \{A, Q, C, R, \mu, S, \bar{y}\}$ and the graphical model of the dynamic texture is shown in Fig. 1(c).

A number of methods are available to learn the parameters of the dynamic texture from a training sequence, including maximum-likelihood methods (e.g., expectation–maximization [24]), non-iterative subspace methods (e.g., N4SID [25], CCA [26], [27]), or a suboptimal, but computationally efficient, least-squares procedure [5]. The dynamic texture has an interesting interpretation when the columns of C are orthogonal (e.g., when learned with the method of [5]). In this case, the columns of C are the principal components of the observations (feature vectors) in time. Hence, the hidden state vector x_t contains the PCA coefficients that generate each observation y_t, where the PCA coefficients (x_t) themselves evolve over time according to a Gauss–Markov process. In this sense, the dynamic texture is an evolving PCA representation of the sequence.

B. Mixture of Dynamic Textures

The DT models a single observed sequence, e.g., an audio fragment lasting several seconds. It could also model multiple sequences, if all exhibited the same dynamic texture (specified by the parameters Θ). However, many applications require the simultaneous analysis of N sequences, where it is known *a priori* that any single sequence exhibits one of a small set of K dynamic textures (with $K \ll N$). For example, the sequences could be audio fragments extracted from a song that can be clustered into a limited number of textures (e.g., corresponding to the verse, chorus, bridge, etc.). Such a clustering would unravel the verse-chorus-bridge structure of the song. An extension of the DT, the DTM model, was proposed in [6] to handle exactly this situation. The DTM is a generative model that treats a *collection* of N sequences as samples from a set of K dynamic textures.

Clustering is performed by first learning a DTM for the sequences, and then assigning each sequence to the DT component with largest posterior probability. This is analogous to clustering feature vectors using a GMM, except that the DTM clusters time-series (sequences of feature vectors), whereas the GMM clusters only feature vectors.

Formally, the DTM [6] is a mixture model where each mixture component is a dynamic texture, and is defined by the system of equations

$$\begin{cases} x_t = A_z x_{t-1} + v_t \\ y_t = C_z x_t + w_t + \bar{y}_z \end{cases} \tag{2}$$

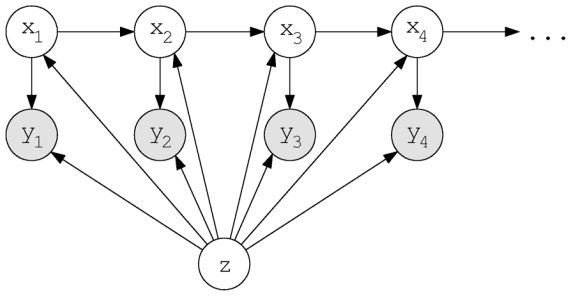

Fig. 2. Graphical model for the dynamic texture mixture. The hidden variable z selects the parameters of the DT represented by the remaining nodes.

where

$$z \sim \text{multinomial}(\alpha_1, \cdots, \alpha_K), \quad \text{s.t.} \sum_{j=1}^{K} \alpha_j = 1 \tag{3}$$

is a random variable that signals the mixture component from which each sequence is drawn. Conditioned on this assignment variable z, the hidden-state x_t, and observation y_t behave like a standard dynamic texture with parameters $\Theta_z = \{A_z, Q_z, C_z, R_z, \mu_z, S_z, \bar{y}_z\}$. The graphical model for the dynamic texture mixture is presented in Fig. 2.

In computer vision, the model has been shown to be a robust model for motion segmentation by clustering patches of video [6]. In this paper, we will use the DTM to segment a song into sections (e.g., verse, chorus, and bridge) in a similar way by clustering audio fragments (sequences of audio feature vectors) extracted from the song. We next present an algorithm for learning the parameters of a DTM from training sequences.

C. Parameter Estimation of DTMs

Given a set of N sequences $\{y^{(i)}\}_{i=1}^N$, where $y^{(i)} = \{y_1^{(i)}, \cdots y_\tau^{(i)}\}$ and τ is the sequence length, the parameters Θ that best fit the observed sequences, in the maximum-likelihood sense [28], can be learned by optimizing

$$\Theta^* = \arg\max_{\Theta} \sum_{i=1}^{N} \log p\left(y^{(i)}; \Theta\right) \tag{4}$$

where $\Theta = \{\Theta_j, \alpha_j\}_{j=1}^K$ are the parameters of the DTM, and $\Theta_j = \{A_j, Q_j, C_j, R_j, \mu_j, S_j, \bar{y}_j\}$ are the parameters for the jth DT component. Note that the data likelihood function $p(y^{(i)}; \Theta)$ depends on two sets of hidden variables: 1) the assignment variable $z^{(i)}$, which assigns each sequence $y^{(i)}$ to a mixture component; and 2) the hidden state sequence $x^{(i)} = \{x_1^{(i)}, \cdots x_\tau^{(i)}\}$ that produces each $y^{(i)}$. Since the data likelihood depends on hidden variables (i.e., missing information), the maximum-likelihood solution of (4) can be found with recourse to the expectation–maximization (EM) algorithm [29]. The EM algorithm is an iterative procedure that alternates between estimating the missing information with the current parameters, and computing new parameters given the estimate of the missing information. For the DTM, each iteration of EM consists of

$$\text{E-Step}: \quad \mathcal{Q}(\Theta; \hat{\Theta}) = \mathrm{E}_{X,Z|Y;\hat{\Theta}}\left(\log p(X, Y, Z; \Theta)\right) \tag{5}$$

$$\text{M-Step}: \quad \hat{\Theta}^* = \arg\max_{\Theta} \mathcal{Q}(\Theta; \hat{\Theta}) \tag{6}$$

where $p(X,Y,Z;\Theta)$ is the complete-data likelihood of the observations $Y = \{y^{(i)}\}_{i=1}^{N}$, hidden states sequences $X = \{x^{(i)}\}_{i=1}^{N}$, and hidden assignment variables $Z = \{z^{(i)}\}_{i=1}^{N}$, parameterized by Θ.

The EM algorithm for the mixture of dynamic textures was derived in [6], and a summary is presented in Algorithm 1. The E-step relies on the Kalman smoothing filter [6], [24] to compute: 1) the expectations of the hidden state variables x_t, given the observed sequence $y^{(i)}$ came from the jth component; and 2) the likelihood of observing $y^{(i)}$ from the jth component. The M-step then computes the maximum-likelihood parameter values for each dynamic texture component j, by averaging over all sequences $\{y^{(i)}\}_{i=1}^{N}$, weighted by the posterior probability of assigning $z^{(i)} = j$.

Algorithm 1 EM for a Mixture of Dynamic Textures

1: **Input:** N sequences $\{y^{(i)}\}_{i=1}^{N}$, number of components K.

2: Initialize $\Theta = \{\Theta_j, \alpha_j\}_{j=1}^{K}$.

3: **repeat**

4: {Expectation Step}

5: **for** $i = \{1,\ldots,N\}$ and $j = \{1,\ldots,K\}$ **do**

6: Compute the conditional expectations

$$\hat{x}_{t|j}^{(i)} = \mathrm{E}_{x^{(i)}|y^{(i)},z^{(i)}=j}\left[x_t^{(i)}\right]$$
$$\hat{P}_{t,t|j}^{(i)} = \mathrm{E}_{x^{(i)}|y^{(i)},z^{(i)}=j}\left[x_t^{(i)}\left(x_t^{(i)}\right)^T\right]$$
$$\hat{P}_{t,t-1|j}^{(i)} = \mathrm{E}_{x^{(i)}|y^{(i)},z^{(i)}=j}\left[x_t^{(i)}\left(x_{t-1}^{(i)}\right)^T\right]$$

by running the Kalman smoothing filter [6], [24] with parameters Θ_j on sequence $y^{(i)}$.

7: Compute the posterior assignment probability

$$\hat{\mathbf{z}}_{i,j} = p\left(z^{(i)} = j | y^{(i)}\right) = \frac{\alpha_j p\left(y^{(i)}|z^{(i)}=j\right)}{\sum_{k=1}^{K}\alpha_k p\left(y^{(i)}|z^{(i)}=k\right)}.$$

8: **end for**

9: {Maximization Step}

10: **for** $j = 1$ to K **do**

11: Compute aggregate expectations

$$\hat{N}_j = \sum_i \hat{\mathbf{z}}_{i,j}, \quad \Phi_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=1}^{\tau}\hat{P}_{t,t|j}^{(i)}$$
$$\xi_j = \sum_i \hat{\mathbf{z}}_{i,j}\hat{x}_{1|j}^{(i)}, \quad \varphi_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=2}^{\tau}\hat{P}_{t,t|j}^{(i)}$$
$$\eta_j = \sum_i \hat{\mathbf{z}}_{i,j}\hat{P}_{1,1|j}^{(i)}, \quad \phi_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=2}^{\tau}\hat{P}_{t-1,t-1|j}^{(i)}$$
$$\Psi_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=2}^{\tau}\hat{P}_{t,t-1|j}^{(i)}$$
$$\Lambda_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=1}^{\tau}\left(y_t^{(i)} - \bar{y}_j\right)\left(y_t^{(i)} - \bar{y}_j\right)^T$$
$$\Gamma_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=1}^{\tau}\left(y_t^{(i)} - \bar{y}_j\right)\left(\hat{x}_{t|j}^{(i)}\right)^T$$
$$\gamma_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=1}^{\tau}y_t^{(i)}, \quad \beta_j = \sum_i \hat{\mathbf{z}}_{i,j}\sum_{t=1}^{\tau}\hat{x}_{t|j}^{(i)}.$$

12: Compute new parameters $\{\Theta_j, \alpha_j\}$

$$C_j^* = \Gamma_j(\Phi_j)^{-1}, \quad R_j^* = \frac{1}{\tau\hat{N}_j}\left(\Lambda_j - C_j^*\Gamma_j\right)$$
$$A_j^* = \Psi_j(\phi_j)^{-1}, \quad Q_j^* = \frac{1}{(\tau-1)\hat{N}_j}\left(\varphi_j - A_j^*\Psi_j^T\right)$$
$$\mu_j^* = \frac{1}{\hat{N}_j}\xi_j, \quad S_j^* = \frac{1}{\hat{N}_j}\eta_j - \mu_j^*\left(\mu_j^*\right)^T$$
$$\alpha_j^* = \frac{\hat{N}_j}{N}, \quad \bar{y}_j^* = \frac{1}{\tau\hat{N}_j}\left(\gamma_j - C_j^*\beta_j\right).$$

13: **end for**

14: **until** convergence

15: **Output:** $\{\Theta_j, \alpha_j\}_{j=1}^{K}$

It is known that the accuracy of parameter estimates produced by EM is dependent on how the algorithm is initialized. We use the initialization strategy from [6], where EM is run several times with an increasing number of mixture components. After each EM converges, one of the components is duplicated and its parameters are perturbed slightly, and EM is run again on the new mixture model. More details on the EM algorithm for DTM and the initialization strategy are available in [6].

IV. Song Segmentation With DTM

Fig. 3 outlines our approach to song segmentation using the DTM model. First, audio features vectors are extracted from the song's audio waveform [e.g., Mel-frequency cepstral coefficients shown in Fig. 3(b)]. Overlapping sequences of audio feature vectors are extracted from a 5 second fragment of the song where the start position of the fragment slides through the entire song with a large step-size (∼0.5 s). A DTM is learned from the collection of these audio fragments and a **coarse** song segmentation is obtained by assigning each 5 second audio fragment to the most probable DTM component [Fig. 3(c)]. Next, we **constrain** the assigned segmentation so that very short segments are unlikely [Fig. 3(d)]. Finally, we run a second segmentation using sequences with a much smaller fragment length (∼1.75 s) and step size (∼0.05 s) to **refine** the precise location of the segment boundaries [Fig. 3(e)] and evaluate the results with reference to a human-labeled "true" segmentation [Fig. 3(f)]. Each of these steps is described in detail below.

A. Features

The content of each 22 050 Hz-sampled, monaural waveform is represented using two types of music information features.

1) Mel-Frequency Cepstral Coefficients: Mel-frequency cepstral coefficients (MFCCs), developed for speech analysis

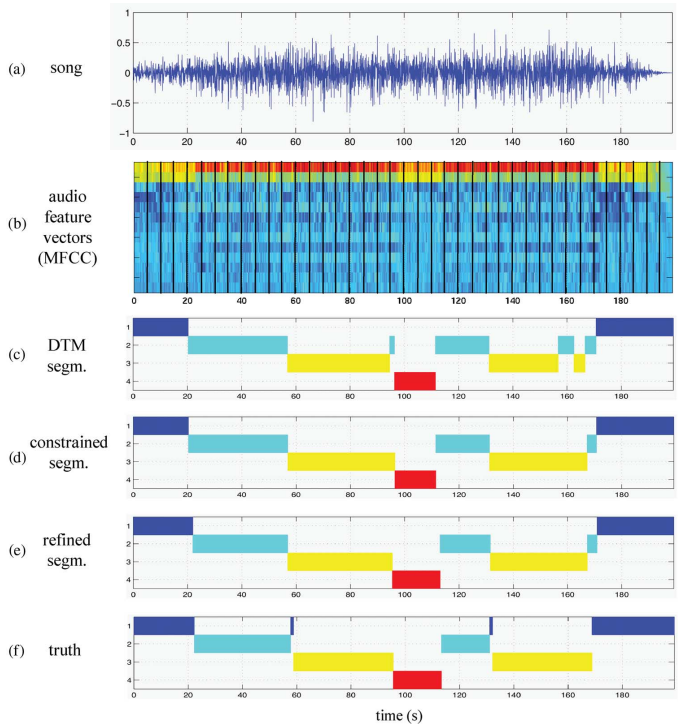

Fig. 3. DTM song segmentation. A song's waveform (a) is represented as a series of audio feature vectors that are collected into short, overlapping sequences (b). These sequences of feature vectors are modeled as a dynamic texture mixture and the song is segmented based on the dynamic texture mixture component to which each sequence is assigned (c). Segments are constrained (d) and refined (e) to produce a final segmentation which is evaluated with reference to a human labeled ground-truth segmentation (f).

[30], describe the timbre or spectral shape of a short-time piece of audio and are a popular feature for a number of music information analysis tasks, including segmentation [3], [7], [19]. We compute the first 13 MFCCs for half-overlapping frames of 256 samples (each feature vector summarizes 12 ms of audio, extracted every 6 ms). In music information retrieval, it is common to augment the MFCC feature vector with its instantaneous first and second derivatives, in order to capture some information about the temporal evolution of the feature. When using the DT, this extra complexity is not required since the temporal evolution is modeled explicitly by the DT.

2) Chroma: Chroma features have also been successfully applied for song segmentation [17], [18], [31]. They represent the harmonic content of a short-time window of audio by computing the spectral energy present at frequencies that correspond to each of the 12 notes and their octave harmonics in a standard chromatic scale. We compute a 12-dimensional chroma feature vector from three-quarter overlapping frames of 2048 samples (each feature vector summarizes 93 ms of audio, extracted every 23 ms).

B. Song Segmentation

Song segmentation is performed with the DTM using a coarse-to-fine approach. A DTM is learned from the collection of audio fragments, using the EM algorithm described in Section III-C. A coarse song segmentation is formed by assigning each fragment to the DTM component with largest posterior probability, i.e.,

$$j^{*(i)} = \arg\max_j \frac{\alpha_j p\left(y^{(i)};\Theta_j\right)}{\sum_{k=1}^{K}\alpha_k p\left(y^{(i)};\Theta_k\right)} \tag{7}$$

where $p(y^{(i)};\Theta_j)$ is the likelihood of sequence $y^{(i)}$ under the jth mixture component Θ_j. Next, musical constraints are applied to the segmentation, and the boundaries are refined for better localization.

C. Musical Constraints on Segments

Levy and Sandler [13] note that musical segments are most likely to last 16 or 32 beats (4 or 8 bars of music in standard 4/4 time). They find that imposing constraints on the minimum segment length results in improved segmentations. To include this constrained clustering in our model, we wish to encourage audio fragments which are close in time to be assigned to the same segment class. This defines a Markov random field (MRF) over the DTM's assignment variables, Z, which restricts the probability that $z^{(i)}$, the class label variable for a given output $y^{(i)} \in Y$, will differ from the labels assigned to sequences neighboring $y^{(i)}$.

The MRF penalizes the class conditional likelihoods output by the DTM in proportion to their disagreement with the class labels assigned to neighboring sequences. The constrained assignments are estimated as in [13] using iterated conditional modes (ICM) as follows. Labels $j^{*(i)}$ are first assigned to all audio fragments, as in (7). Next, the constraints are incorporated while iterating through each fragment i. The log-likelihood, with constraints, of fragment i under each mixture component j is computed

$$\log\tilde{p}\left(y^{(i)};\Theta_j\right) = \log p\left(y^{(i)};\Theta_j\right) - \sum_{c=-W/2,c\neq 0}^{W/2}\phi_j\left(j^{*(i+c)}\right) \tag{8}$$

where W is the length of the temporal neighborhood surrounding the i^{th} fragment over which the constraints are imposed, and

$$\phi_j\left(j^{*(c)}\right) = \begin{cases} 0, & \text{if } j^{*(c)} = j \\ \lambda, & \text{otherwise} \end{cases} \tag{9}$$

adds a penalty of λ when the neighboring class labels $j^{*(c)}$ do not match the current label j. The new constrained class label of the fragment $j^{*(i)}$ is then assigned according to

$$j^{*(i)} = \arg\max_j \log\tilde{p}\left(y^{(i)};\Theta_j\right). \tag{10}$$

The process is iterated, for all i, until convergence of the class labels for all fragments. The fixed cost parameter λ and the neighborhood size W over which the constraints are imposed are determined experimentally and depend on the type of feature and sequence step size being used. We find that $\lambda \approx 90$ and a constraint neighborhood corresponding to 15–20 seconds is optimal.

D. Refining Segment Boundaries

This first segmentation is relatively coarse and can localize segment boundaries, at best, to within 0.25 seconds, due to the

large step size and the poor localization properties of using long audio fragments. Precise boundaries are found by extracting audio fragments with shorter length ($\sim$1.75 s) and step size ($\sim$0.05 s). We assign these short fragments to the *same* DTM components learned in Section IV-B, resulting in a finer segmentation of the song. This tends to over-segment songs as the DTM state changes too frequently: the coarse segmentation more accurately learns the temporal structure of each song. However, we can refine the original, coarse segmentation by moving each segment boundary to the closest corresponding boundary from the fine segmentation. These refined boundaries are likely to be valid since they were produced by the same DTM model. They are expected to provide a more precise estimate of the true segment boundaries.

V. Segmentation Evaluation

In this section, we evaluate the proposed algorithm for song segmentation on two music datasets. We also test the applicability of the algorithm to the similar task of music boundary detection.

A. Data

We evaluate the automatic song segmentation performance of the DTM model on two separate musical datasets for which human-derived structural segmentations exist.

1) RWC Dataset: The RWC Music Database (RWCMDB-P-2001) [32] contains 100 Japanese pop songs where each song has been segmented into coherent parts by a human listener [33]. The segments are accurate to 10 ms and are labeled with great detail. For this work we group the labeled segments into four possible classes: "verse" (i.e., including verse A, verse B, etc.), "chorus," "bridge," and "other" ("other" includes labels such as "intro," "ending," "pre-chorus," etc. and is also used to model any silent parts of the song). This results in a "ground truth" segmentation of each song with four possible segment classes. On average, each song contains 11 segments (with an average segment length of 18.3 s).

2) PopMusic Dataset: The second dataset is a collection of 60 popular songs from multiple genres including rock, pop and hip-hop. Half the tracks are by the Beatles and the remainder are from a selection of popular artists from the past 40 years including Radiohead, Michael Jackson and the Beastie Boys. The human segmentations for this dataset were used by Levy and Sandler [13] to evaluate their musical segmentation algorithm. The ground truth segmentation of each song contains between 2 and 15 different segment classes (mean $= 6.3$) and, on average, each song also contains 11 segments (with an average segment length of 16.5 s).

B. Experimental Setup

The songs in the RWC dataset were segmented with the DTM model into $K = 4$ segments (chosen to model "verse," "chorus," "bridge," and "other" segments and for comparison to previous work on the same dataset [19]) using the method described in Section IV. DTM models trained using either the MFCC or chroma features, we denote DTM-MFCC and DTM-Chroma, respectively. For DTM-MFCC, we use a sequence length of 900 MFCC feature vectors (extracted from 5.2 s of audio content) and a step-size of 100 feature frames, while for DTM-Chroma, we use a sequence length of 600 chroma feature vectors (13.9 s of audio) and a step-size of 20 frames. The dimension of the hidden state-space of the DTM was $n = 7$ for MFCC, and $n = 6$ for chroma.

TABLE I
Song Segmentation of the RWC Dataset

Model	Rand Ind	Pairwise F	Av. Segments
DTM-MFCC	0.75	0.62	10.8
DTM-Chroma	0.73	0.58	11.9
GMM-MFCC	0.66	0.52	58.7
GMM-Chroma	0.61	0.51	26.3
Constant	0.32	0.48	1
Random	0.57	0.32	279.0
Truth	1.00	1.00	11

For comparison, we also segment the songs using a GMM trained on the same feature data [3]. We learn a $K = 4$ component GMM for each song, and segment by assigning features to the most likely Gaussian component. Since segmentation decisions are now made at the short time-scale of individual feature vectors, we smooth the GMM segmentation with a length-1000 maximum-vote filter. We compare these models against two baselines: "constant" assigns all windows to a single segment, "random" selects segment labels for each window at random.

We quantitatively measure the correctness of a segmentation by comparing with the ground-truth using two clustering metrics: 1) the Rand index [34] intuitively corresponds to the probability that any pair of audio fragments will be clustered correctly, with respect to each other (i.e., in the same cluster, or in different clusters); 2) the pairwise F-measure [13] compares pairs of feature sequences that the model labels as belonging to the same segment-type with the true segmentation. If P_m is the set of audio fragment pairs that the model labels as similar and P_h is the set of fragment pairs that the human segmentation indicates should be similar then

$$\text{pairwise precision, } P_{\text{pairwise}} = \frac{|P_m \cap P_h|}{|P_m|}$$

$$\text{pairwise recall, } R_{\text{pairwise}} = \frac{|P_m \cap P_h|}{|P_h|}$$

$$\text{pairwise F-measure} = \frac{2 * P_{\text{pairwise}} * R_{\text{pairwise}}}{P_{\text{pairwise}} + R_{\text{pairwise}}}.$$

We also report the average number of segments per song.

C. Segmentation Results

Table I reports the segmentation results on the RWC dataset. DTM-MFCC outperforms all other models, with a Rand index of 0.75[1] and a pairwise F-measure of 0.62. GMM performs significantly worse than DTM, e.g., the F-measure drops to 0.52 on the MFCC features. In particular, the GMM grossly over-segments the songs, leading to very low pairwise precision. This suggests that there is indeed a benefit in modeling the temporal dynamics with the DTM.

For the PopMusic dataset, we no longer restrict the segmentation to just four classes and instead attempt to model all possible

[1]This Rand index result is slightly lower than the value reported in [4] as, in the current work, we allow each model segment to match only one reference segment. This is consistent with the evaluations in [13].

TABLE II
SONG SEGMENTATION OF THE POPMUSIC DATASET

Model	Rand Ind.	Pairwise F	Av. Segments
DTM-MFCC	0.78	0.62	10.7
DTM-Chroma	0.74	0.51	12.0
GMM-MFCC	0.72	0.49	78.9
GMM-Chroma	0.67	0.50	32.4
Constant	0.32	0.48	1
Random	0.57	0.32	279.0
Truth	1.00	1.00	11.1

TABLE III
EFFECT OF MUSICAL CONSTRAINTS AND BOUNDARY REFINEMENT ON DTM-MFCC SEGMENTATION OF THE POPMUSIC DATASET

Model	Rand Ind.	Pairwise F	Av. Segments
Coarse	0.762	0.577	17.9
Refine	0.770	0.587	17.9
Constrain	0.773	0.614	10.7
Constrain+Refine	0.779	0.620	10.7

segment classes. Given that each song in the dataset has an average of 6.3 different segments, we set the number of mixture components in the DTM model $K = 6$ and increase the state-space dimension to $n = 12$. The segmentation results are shown in Table II and are very similar to the results obtained for the RWC dataset. We note that the DTM-MFCC model F-measure of $0.6196 +/- 0.0163$ improves on the segmentation algorithm of Levy and Sandler [13] who report an average F-measure of 0.603 using $K = 6$ clusters to segment the same data. The result of [13] lies at the minimum of our confidence interval. A paired comparison of the results for each song would be required to conclusively determine the significance of our improvement, but this data was not available in [13]. The state-of-the-art segmentation performance validates the DTM's capacity to model musical audio content and its promise for applications beyond segmentation, as a general, generative model for music.

Looking at the different feature representations, the DTM-MFCC outperforms DTM-Chroma on both datasets, with F-scores of 0.62 versus 0.58 on RWC, and 0.62 versus 0.51 on PopMusic. On the other hand, GMM-MFCC and GMM-Chroma perform similarly (F-scores of 0.52 versus 0.51 on RWC, and 0.49 versus. 0.50 on Pop). These results suggest that chroma time-series are not as well modeled as MFCC time–series by the DTM model. In particular, each coordinate of the chroma feature vector is active (nonzero) when a particular musical key is present, and hence the time-series of chroma features will tend to be "spiky," depending on when the chords change in the song. The chroma features are also non-negative. Because of these two aspects, the chroma time-series is not as well modeled by the DTM, which is better suited for modeling second-order smooth time-series with Gaussian noise.

Table III examines the impact of the musical constraints and boundary refinement on the segmentations produced by our best model, the DTM-MFCC model. We see that the musical constraints improve the final segmentation of the PopMusic dataset by removing short, inaccurate segments and thus reducing the overall number of segments (the average number of segments drops from 17.9 to 10.7 where the true segmentations contain

[2]http://cosmal.ucsd.edu/cal/projects/segment/

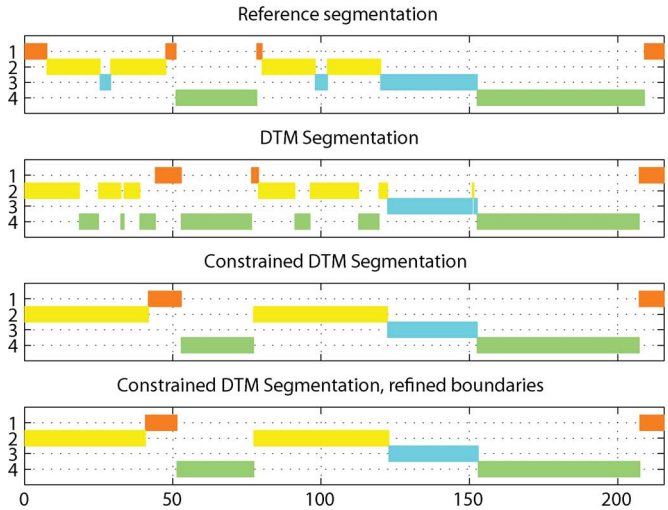

Fig. 4. DTM segmentations and reference segmentation of the track "p053" from the RWC dataset (Rand Index $= 0.78$, Pairwise F $= 0.66$). The addition of the musical constraints removes short segments.

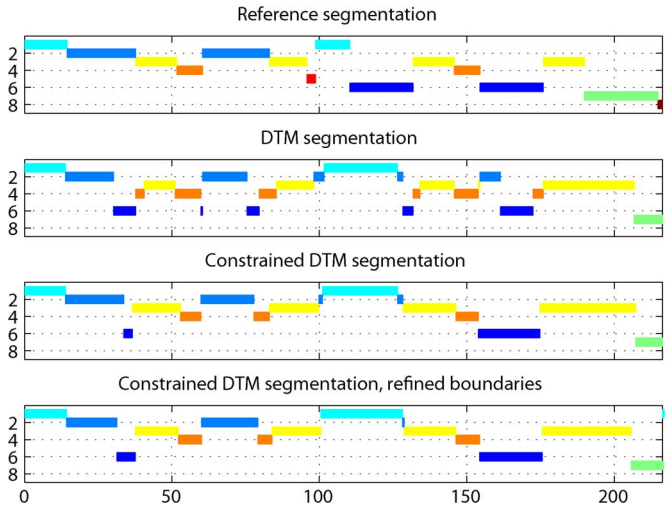

Fig. 5. DTM segmentations and reference segmentation of "It's Oh So Quiet" by Björk from the PopMusic dataset (Rand Index $= 0.82$, Pairwise F $= 0.55$). When there are more classes in the reference segmentation than there are DTM components, the model successfully ignores the smallest classes (classes 5 and 8 in this example).

an average of 11.1 segments). Indeed, these constraints often remove certain segment classes from the output altogether. In cases where the true segmentation had less than K different classes, the model can now ignore irrelevant classes.

Examples of DTM song segmentations are compared to the ground truth in Figs. 4 and 5 (more examples available online[2]). We see that, while most DTM segments are accurate, there are a few errors due to imprecise borders, and some cases where the model over- or under-segments.

D. Boundary Detection Results

In addition to evaluating the segmentation performance of the DTM model, we can consider its accuracy in detecting the boundaries between segments (without trying to label the segment classes). We evaluate boundary detection performance using two median time metrics: true-to-guess (T-to-G) and guess-to-true (G-to-T), respectively, measure the median time from each true boundary to the closest model estimate, and

TABLE IV
DTM BOUNDARY DETECTION PERFORMANCE ON THE RWC DATASET, COMPARED TO A COMMERCIAL ONLINE SERVICE "THE ECHONEST" AND THE SUPERVISED METHOD OF [19]

Model	G-to-T	T-to-G	P	R	F
DTM-MFCC	3.21	2.96	0.22	0.22	0.22
DTM-Chroma	5.69	4.46	0.10	0.12	0.11
EchoNest	5.08	1.84	0.18	0.21	0.19
[19]	4.29	1.82	0.33	0.46	0.38

TABLE V
DTM BOUNDARY DETECTION PERFORMANCE ON THE POPMUSIC DATASET COMPARED TO ECHONEST

Model	G-to-T	T-to-G	P	R	F
DTM-MFCC	3.58	2.99	0.62	0.65	0.61
DTM-Chroma	5.82	4.39	0.41	0.46	0.42
EchoNest	5.59	5.32	0.41	0.56	0.45

the median time from each model estimate to the closest true boundary, as in [19]. We also consider the precision, recall and F-measure of boundary detection where a boundary output by the model is considered a "hit" if it is within a certain time threshold of a true segment boundary, as in [13], [19], and [20].

The boundary detection results, averaged over the 100 RWC songs $(K = 4)$, are presented in Table IV. We use a threshold of 0.5 s for comparison to [19], who tackle the boundary detection problem by learning a supervised classifier that is optimized for boundary detection. In Table V, we show results for the PopMusic dataset $(K = 6)$ where we now use a hit threshold of 3 s, following [13] and [20]. For both datasets, we also compare with the music analysis company EchoNest [35], which offers an online service for automatically detecting music boundaries.

For the PopMusic dataset, the boundary detection results for the DTM segmentation (boundary F-measure $= 0.61$) are comparable to the performance of Levy and Sandler's segmentation algorithm (best boundary F-measure $= 0.604$) [13]. However, neither system approaches the accuracy of specialized boundary detection algorithms (e.g., Ong and Herrera [20] achieve boundary F-measure of 0.75 on a test set of similar Beatles music). Boundary detection algorithms (e.g., [19], [20]) are designed to detect novelty between successive feature frames or respond to musical cues such as drum fills or changes in instrumentation which indicate that one segment is ending and another beginning. However, they do not model the musical structure and there is no characterization of the segments between the boundaries as the DTM or [13] provides. In future work, we will investigate using a supervised boundary detection algorithm to improve on the simple refinement of the DTM segmentation that we propose in Section IV-D.

VI. APPLICATIONS OF AUTOMATIC SONG SEGMENTATION

In this section, we demonstrate several applications of the automatic song segmentation algorithm to music annotation, retrieval, and visualization.

A. Autotagging Song Segments

A number of algorithms have been proposed for automatically associating music content with descriptive semantic phrases or "tags" [1], [36], [37]. These supervised methods use large corpora of semantically tagged music to discover patterns in the audio content that are correlated with specific tags. Various methods exist for collecting the tags used to train these systems including hiring human subjects to label songs [1], mining websites [38], or online games [39] (see [40] for a review of the performance of each of these methods).

The tags generated by most of these method are presumed to be associated with the entire song. However, depending on the specific tag and the source from which it was collected, this may not be true. For example, the song "Bohemian Rhapsody" by Queen might accurately be tagged by one listener as a "melancholy piano ballad," another listener might refer to the "energetic opera with falsetto vocal harmonies," while a third listener might hear "screaming classic rock with a powerful electric guitar riff." The training of autotagging algorithms [1], [36] is designed to accommodate the fact that not all of the features present in the labeled music audio content will actually manifest the associated tags. However, this "multiple instance learning" problem presents a challenge for evaluating the output of such algorithms since many of the tags apply to only certain segments of the song.

The solution to the "Bohemian Rhapsody problem" lies in first dividing a song into musically homogeneous segments and then tagging each of the segments individually. We use the music tagging algorithm described in [1] to associate the segments extracted from the 60-song PopMusic dataset described in Section V with 149 semantic tags from the CAL-500 vocabulary used in [1]. Given a music waveform, the output of this algorithm is a *semantic multinomial distribution*, a vector of probabilities that each tag in the vocabulary applies to the music content. These tags include genre, emotion, instrument, vocal style and song-usage descriptors. The accuracy of the tagging algorithm has been found to predict one human's responses as accurately as another human would [1] (i.e., it approaches the limit imposed by musical subjectivity) and was the best performing automatic music tagging algorithm in the 2008 Music Information Retrieval Evaluation eXchange (MIREX) contest [41].

Fig. 6 demonstrates the $K = 6$ class DTM segmentation of the song "Bohemian Rhapsody." Four of the top automatically determined tags are displayed for each segment where the first indicates the segment's most likely genre, the second detects the most prevalent instrument or vocal characteristic, the third describes the emotion evoked by the segment, and the fourth gives a general description of the segment. The majority of the tags accurately describe the musical content although a few are clearly incorrect (e.g., there is no saxophone in the second segment and, though his voice was high pitched, Freddie Mercury was not a female singer!). More importantly, there is a big difference between the tags that describe the mellow, acoustic, early segments of the song and those used to describe the more rocking, up-tempo segments towards the end. Compare the tags for each segment in Fig. 6 with the top tags output for the entire song which generically describe Bohemian Rhapsody as a *pop* song with a *female vocal* that is *pleasant* and is *not very danceable*.

Table VI further illustrates the need for segmentation before semantic analysis of audio content. In the left column, we present the average Kullback–Leibler (KL) divergence between

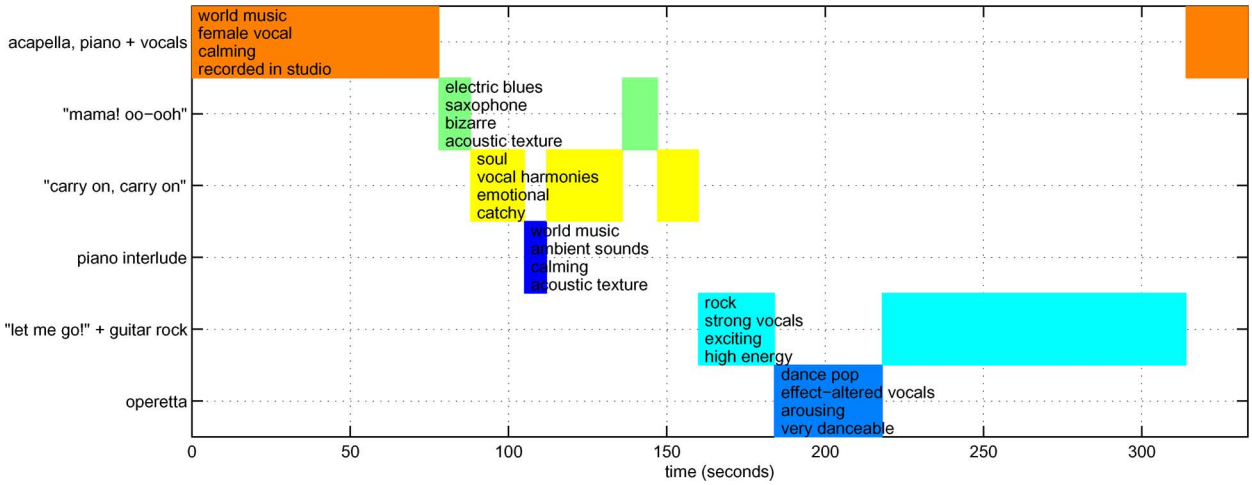

Fig. 6. DTM segmentation of the song "Bohemian Rhapsody" by Queen. The automatically generated tags show the most likely genre, the most prevalent instrument or vocal characteristic, the emotion evoked and a general description of each segment class. Treating the song as a whole results in the general tags *pop*, *female vocal*, *pleasant*, and *not very danceable*. The y-axis labels are added by the authors to highlight the musical or lyrical content of each segment class.

TABLE VI
MEAN SEMANTIC KL DIVERGENCE AND TEMPO MISMATCH BETWEEN A DTM SEGMENT AND ANOTHER SEGMENT FROM THE SAME CLASS, FROM THE SAME SONG (BUT A DIFFERENT CLASS) AND FROM A DIFFERENT SONG, AVERAGED OVER ALL SONGS FROM THE POPMUSIC DATASET. SECTION VI-B EXPLAINS THE SIMILAR DT (BOTTOM ROW)

	KL divergence	Tempo mismatch
Same Class	0.04	0.20
Same Song	0.54	0.29
Different Song	0.70	0.49
Similar DT	0.33	0.29

the semantic multinomial describing a single, automatically-extracted segment of a given song from the PopMusic dataset (e.g., the first chorus of song 1) and other segments from that song that are assigned to the same DTM component (e.g., other choruses from song 1), segments from the same song but different classes (e.g., verse, bridge, etc. from song 1) and segments chosen randomly from any other song in the dataset, averaged over all songs from the PopMusic dataset. This method of using semantic descriptors to determine audio similarity has been shown to be more accurate than calculating similarity of the acoustic content directly [2]. Table VI demonstrates that while segments assigned to the same DTM components produce almost identical semantic descriptions $(\mathrm{KL} = 0.04)$, there is a large divergence between the semantic multinomial distributions of segments from different DTM components from within the same song $(\mathrm{KL} = 0.54)$, approaching the divergence between two random segments $(\mathrm{KL} = 0.70)$.

The right column of Table VI presents the average tempo mismatch between segments, averaged over all songs from the PopMusic dataset. We use an automatic tempo extraction algorithm [42] to compute the tempo, in beats-per-minute (bpm), of each segment. As in [43], we deem two segments to have similar tempi if the bpm of the second is within $\pm 4\%$ of the bpm of the first, where, to account for confusion in the meter, matches with one-third, half, double or triple the first bpm are also permitted. We see that segments from the same class differ in tempo 20% of the time whereas two random segments have almost 50% chance of a tempo mismatch. The average tempo mismatch between segments from the same class in the true segmentation is 10%. These results suggest that the DT is also capturing temporal information, along with the semantic information.

B. Song Segment Retrieval

The segmentation of a song obtained by modeling a series of coherent audio fragments with a dynamic texture can be used to retrieve musically similar segments from different songs. We can now answer questions like "what sounds similar to the verse of this song?" We represent each segment by its corresponding dynamic texture component in the DTM-MFCC model and measure similarities between dynamic textures with the KL divergence between them [22] (note that this KL divergence is now between dynamic texture models, rather than the KL between semantic multinomial distributions considered in the previous section and presented in Table VI). Using each song segment from the RWC dataset as a query, the five closest retrieved segments are presented online.[3] Qualitatively, the retrieved segments are similar in both audio texture and temporal characteristics. For example, a segment with slow piano will retrieve other slow piano songs, whereas a rock song with piano will retrieve more upbeat segments.

To quantitatively evaluate the song segment retrieval, we compute the average semantic KL divergence and tempo mismatch between each query segment and the retrieved song segments that are modeled with the most similar dynamic texture component. The results for the single most similar DT are presented in the bottom row of Table VI. It can be seen that two segments with most similar DT components are, on average, more semantically similar than two segments from the same song (KL of 0.33 versus 0.54). The tempo mismatch between retrieved segments is the same as segments from the same song but significantly lower than segments from different songs (note that 75% of the most similar retrieved segments came from the same song as the query—DT components of the same DTM model). This indicates that the dynamic texture model captures both the timbre of the audio content, evidenced by the similar semantic descriptions (derived from analysis of the instantaneous spectral characteristics), as well as temporal characteristics, as shown by the similar tempi.

[3]http://cosmal.ucsd.edu/cal/projects/segment/

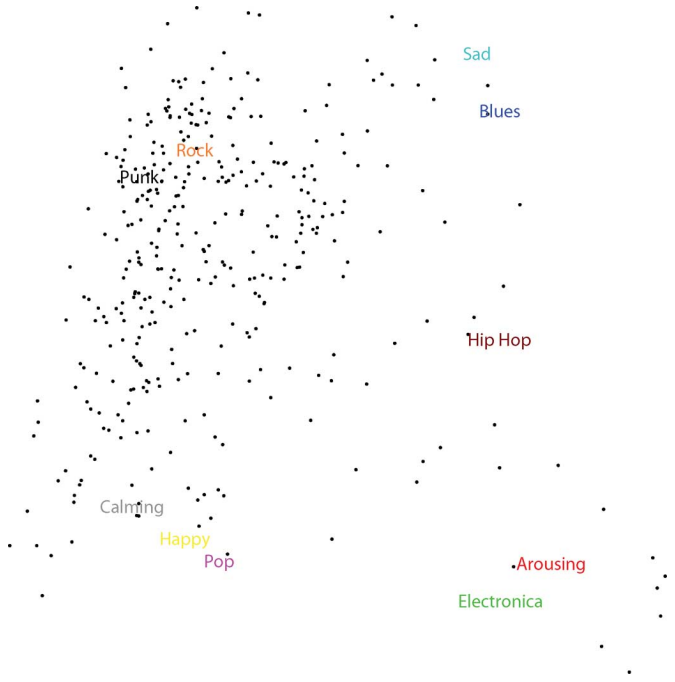

Fig. 7. 2-D visualization of the distribution of song segments. Each black dot is a song segment. Areas of the space are automatically tagged based on the system described in Section VI-A.

In order to visualize the distribution of songs in the dataset, the automatically-extracted segments of songs from the Pop-Music dataset were embedded into a 3-D manifold using local-linear embedding (LLE) [44] of the KL similarity matrix computed above for song retrieval. Two dimensions of the embedding are shown in Fig. 7.

We add interpretability to this embedding by inferring genre and emotion tags that best describe each part of the space. For each tag, we compute a kernel density estimate of the tag's probability distribution by placing a Gaussian kernel at each segment point in the embedding space. We weight each kernel by the tag probability assigned to the corresponding segment by the autotagging algorithm described in Section VI-A. The result is an estimate of the distribution over the embedding space of a each tag's relevance. In Fig. 7, we label the embedding space by finding the centroid of the area of the top 20% of each of these probability densities.

The four emotion tags in Fig. 7 illustrate that the largest variance in the DTM segments results in good separation between the tags "happy" and "sad" and between "calming" and "arousing," corresponding with the psychological primitives or "core affect" described in [45]. The six genre tags show a progression from synthesized music like "hip hop" and "electronica" in the lower right, through "blues" and "pop" in the center to "rock" and "punk" at the top left. This automatic labeling of the embedding space again suggests that the DTM model is successfully capturing both the audio texture (e.g., separating happy and sad) and the temporal characteristics (e.g., separating calming and arousing) of the songs.

VII. Conclusion

We have presented a new representation for musical audio, the dynamic texture (DT), which simultaneously accounts for both the instantaneous content of short audio fragments as well as the evolution of the audio over time. We applied the new representation to the task of song segmentation (i.e., automatically dividing a song into coherent segments that human listeners would label as verse, chorus, bridge, etc.), by modeling audio fragments from a song as samples from a DTM model. Experimentally, the resulting segmentation algorithm achieves state-of-the-art results in segmentation experiments on two music datasets. More importantly, the generative nature of the proposed model of music makes it directly applicable to a wider and more diverse range of applications, compared to algorithms specifically developed for music segmentation. Its state-of-the-art results on music segmentation indicate that the dynamic texture representation shows promise as a new model for automatic music analysis. Future work will consider using the DTM model to move beyond the bag-of-features representation in applications such as music similarity and automatic music tagging.

Another interesting direction for future work is to use more complex "switching" DT models [46]–[48] to improve on the DTM segmentation. These models should better localize the segment boundaries, as they operate on the entire song, rather than in a fragment-based manner. In general, these switching models are more difficult to learn robustly, due to the complexity of the models and the necessity for approximate inference. However, their effectiveness can be greatly increased by initializing the learning algorithm with a good segmentation, such as the one provided by the proposed DTM segmentation algorithm. Also a potential direction of future work is to modify the DTM so that it better models the properties of the chroma time-series.

Acknowledgment

This research utilized the AIST Annotation for the RWC Music Database (Popular Music Database) and the Queen Mary reference structural segmentations.

References

[1] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Trans. Acoust., Speech, Lang. Process.*, vol. 16, no. 2, pp. 467–476, Feb. 2008.

[2] L. Barrington, A. Chan, D. Turnbull, and G. Lanckriet, "Audio information retrieval using semantic similarity," in *Proc. Proc. IEEE ICASSP*, 2007, vol. 2, pp. 725–728.

[3] J.-J. Aucouturier, F. Pachet, and M. Sandler, "'The Way It Sounds': Timbre models for analysis and retrieval of music signals," *IEEE Trans. Multimedia*, vol. 7, no. 6, pp. 1028–1035, Dec. 2005.

[4] L. Barrington, A. B. Chan, and G. Lanckriet, "Dynamic texture models of music," in *Proc. IEEE ICASSP*, 2009, pp. 1589–1592.

[5] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *Int. J. Comput. Vis.*, vol. 51, no. 2, pp. 91–109, 2003.

[6] 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.

[7] J. Foote, "Visualizing music and audio using self-similarity," in *Proc. Int. Multimedia Conf.*, 1999, pp. 77–80.

[8] J. Paulus and A. Klapuri, "Music structure analysis using a probabilistic fitness measure and an integrated musicological model," in *Proc. 9th Conf. Music Inf. Retrieval (ISMIR)*, 2008, pp. 369–374.

[9] M. Levy, M. Sandler, and M. Casey, "Extraction of high-level musical structure from audio data and its application to thumbnail generation," in *Proc. IEEE ICASSP*, May 2006, vol. 5, p. V-V.

[10] B. Logan and S. Chu, "Music summarization using key phrases," in *Proc. IEEE ICASSP*, 2000, pp. 749–752.

[11] J. Reed and C. Lee, "A study on music genre classification based on universal acoustic models," in *Proc. 7th Conf. Music Information Retrieval (ISMIR)*, 2006.

[12] S. Abdallah, M. Sandler, C. Rhodes, and M. Casey, "Using duration models to reduce fragmentation in audio segmentation," *Mach. Learn.: Special Iss. Mach. Learn. in and for Music*, vol. 65, no. 2–3, pp. 485–515, Dec. 2006.
[13] M. Levy and M. Sandler, "Structural segmentation of musical audio by constrained clustering," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 318–326, Feb. 2008.
[14] G. Peeters, A. Burthe, and X. Rodet, "Toward automatic music audio summary generation from signal analysis," in *Proc. 3rd Conf. Music Inf. Retrieval (ISMIR)*, 2002, pp. 94–100.
[15] W. Chai and B. Vercoe, "Music thumbnailing via structural analysis," in *Proc. 11th ACM Int. Conf. Multimedia*, 2003, pp. 223–226.
[16] J. Hsu, C. Liu, and L. Chen, "Discovering nontrivial repeating patterns in music data," *IEEE Trans. Multimedia*, vol. 3, no. 3, pp. 311–325, Sep. 2001.
[17] M. Bartsch and G. Wakefield, "To catch a chorus: Using chroma-based representations for audio thumbnailing," in *Proc. IEEE Workshop Applicat. Signal Process. Audio Acoust.*, 2001, pp. 15–19.
[18] M. Goto, "A chorus-section detection method for musical audio singals and its application to a music listening station," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 1, pp. 1783–1794, Jan., 2006.
[19] D. Turnbull, G. Lanckriet, E. Pampalk, and M. Goto, "A supervised approach for detecting boundaries in music using difference features and boosting," in *Proc. 8th Conf. Music Inf. Retrieval (ISMIR)*, 2007.
[20] B. Ong and P. Herrera, "Semantic segmentation of music audio contents," in *Proc. Int. Comput. Music Conf. (ICMC)*, 2005.
[21] P. Saisan, G. Doretto, Y. Wu, and S. Soatto, "Dynamic texture recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognition*, 2001, vol. 2, pp. 58–63.
[22] A. B. Chan and N. Vasconcelos, "Probabilistic kernels for the classification of auto-regressive visual processes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognition*, 2005, vol. 1, pp. 846–851.
[23] G. Doretto, D. Cremers, P. Favaro, and S. Soatto, "Dynamic texture segmentation," in *Proc. IEEE ICCV*, 2003, vol. 2, pp. 1236–1242.
[24] 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.
[25] P. V. Overschee and B. D. Moor, "N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems," *Automatica*, vol. 30, pp. 75–93, 1994.
[26] W. E. Larimore, "Canonical variate analysis in identification, filtering, and adaptive control," in *Proc. IEEE Conf. Decision Control*, 1990, vol. 2, pp. 596–604.
[27] D. Bauer, "Comparing the CCA subspace method to pseudo maximum likelihood methods in the case of no exogenous inputs," *J. Time Series Anal.*, vol. 26, pp. 631–668, 2005.
[28] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[29] 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.
[30] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[31] B. Ong, E. Gómez, and S. Streich, "Automatic extraction of musical structure using pitch class distribution features," in *Workshop Learning the Semantics of Audio Signals*, 2006.
[32] M. Goto, "Development of the RWC music database," in *Proc. Int. Congr. Acoust.*, April 2004, pp. 553–556.
[33] M. Goto, "AIST annotation for RWC music database," in *Proc. 7th Int. Conf. Music Inf. Retrieval (ISMIR)*, Oct. 2006, pp. 359–360.
[34] L. Hubert and P. Arabie, "Comparing partitions," *J. Classification*, vol. 2, pp. 193–218, 1985.
[35] "EchoNest." [Online]. Available: http://the.echonest.com
[36] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green, "Automatic generation of social tags for music recommendation," *Adv. Neural Inf. Process. Syst.* , vol. 20, pp. 385–392, 2007.
[37] B. Whitman and D. Ellis, "Automatic record reviews," in *Proc. 5th Conf. Music Inf. Retrieval (ISMIR)*, 2004.
[38] P. Knees, T. Pohle, M. Schedl, and G. Widmer, "A music search engine built upon audio-based and web-based similarity measures," in *Proc. ACM SIGIR*, 2007, pp. 447–454.
[39] D. Turnbull, R. Liu, L. Barrington, and G. Lanckriet, "Using games to collect semantic information about music," in *Proc. 8th Conf. Music Inf. Retrieval (ISMIR)*, 2007.
[40] D. Turnbull, L. Barrington, and G. Lanckriet, "Five approaches to collecting tags for music," in *Proc. 9th Conf. Music Inf. Retrieval (ISMIR)*, 2008, pp. 225–230.
[41] L. Barrington, D. Turnbull, and G. Lanckriet, "Auto-tagging music content with semantic multinomials." [Online]. Available: http://www.music-ir.org/mirex/2008/abs/AT_barrington.pdf Oct. 2008
[42] S. Dixon, "Mirex 2006 audio beat tracking evaluation: Beatroot." [Online]. Available: http://www.elec.qmul.ac.uk/people/simond/pub/2006/mirex-beat.pdf 2006
[43] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 5, pp. 1832–1844, Sep. 2006.
[44] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
[45] J. Russell, "Core affect and the psychological construction of emotion," *Psychol. Rev.*, vol. 110, no. 1, pp. 145–172, 2003.
[46] Z. Ghahramani and G. E. Hinton, "Variational learning for switching state-space models," *Neural Comput.*, vol. 12, no. 4, pp. 831–864, 2000.
[47] A. B. Chan and N. Vasconcelos, "Layered dynamic textures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1862–1879, Oct. 2009.
[48] A. B. Chan and N. Vasconcelos, "Variational layered dynamic textures," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognition*, 2009.

Luke Barrington (M'09) received the B.E. (elec.) degree from University College Dublin (UCD), Dublin, Ireland, in 2001 and the M.S. degree from the University of California, San Diego (UCSD) in 2004. He is currently pursuing the Ph.D. degree with a thesis titled "Machines that understand music" in the Electrical and Computer Engineering Department, UCSD.

He is a cofounder of the UCSD Computer Audition Laboratory.

Mr. Barrington received the UCD Young Engineer of the year in 2001. In 2005, he was a National Science Foundation (NSF) EAPSI fellow in Japan. From 2006 to 2008, he was the recipient of a U.S. NSF IGERT Fellowship. He is an avid musician and wails on the guitar.

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), 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.

From 2006 to 2008, he was the recipient of a NSF IGERT Fellowship. 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 at the University of California, San Diego, where he heads the Computer Audition Laboratory.

Prof. Lanckriet was awarded the SIAM Optimization Prize in 2008 and is the recipient of a Hellman Fellowship in 2009. His research focuses on the interplay of convex optimization, machine learning, and signal processing, with applications in computer music and computer audition.