Segmentation, Indexing, and Retrieval for Environmental and Natural Sounds

Gordon Wichern, *Student Member, IEEE*, Jiachen Xue, Harvey Thornburg, *Member, IEEE*, Brandon Mechtley, *Student Member, IEEE*, and Andreas Spanias, *Fellow, IEEE*

***Abstract*—We propose a method for characterizing sound activity in fixed spaces through segmentation, indexing, and retrieval of continuous audio recordings. Regarding *segmentation*, we present a dynamic Bayesian network (DBN) that jointly infers onsets and end times of the most prominent sound events in the space, along with an extension of the algorithm for covering large spaces with distributed microphone arrays. Each segmented sound event is *indexed* with a hidden Markov model (HMM) that models the distribution of example-based queries that a user would employ to *retrieve* the event (or similar events). In order to increase the efficiency of the retrieval search, we recursively apply a modified spectral clustering algorithm to group similar sound events based on the distance between their corresponding HMMs. We then conduct a formal user study to obtain the relevancy decisions necessary for evaluation of our retrieval algorithm on both automatically and manually segmented sound clips. Furthermore, our segmentation and retrieval algorithms are shown to be effective in both quiet indoor and noisy outdoor recording conditions.**

***Index Terms*—Acoustic signal analysis, acoustic signal detection, Bayes procedures, clustering methods, database query processing.**

I. Introduction

CHARACTERIZING sound activity in fixed spaces has proven essential to fields such as anthropology [1], [2] and human perception [3], with recent applications in surveillance [4] and urban planning [5]. There also has been much interest in Continuous Archival and Retrieval of Personal Experience (CARPE), where personal archives of human interactions in both physical and virtual worlds are used to aid memory and other aspects of cognition [6], [7]. Before these archives can be considered usable, however, users need to understand where sound events occur, and they must be able to access all sounds of a given type. That is, there remain key technical challenges regarding *segmentation* (determining where sound events begin or end), *indexing* (computing and storing sufficient information along with each event to distinguish it from other events), and *retrieval* (obtaining all events of a given type). These challenges have received much attention for musical sounds and music information databases, while natural and environmental sounds are often overlooked.

Manuscript received January 06, 2008; revised December 02, 2009. Current version published February 10, 2010. This work was supported by the National Science Foundation under Grants NSF IGERT DGE-05-04647 and NSF CISE Research Infrastructure 04-03428. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Bertrand David.

The authors are with the School of Arts, Media, and Engineering, Arizona State University, Tempe, AZ 85282 USA (e-mail: gordon.wichern@asu.edu; jcxue@asu.edu; harvey.thornburg@asu.edu; bmechtley@asu.edu; spanias@asu.edu).

Digital Object Identifier 10.1109/TASL.2010.2041384

A. Segmentation

Regarding segmentation, our primary objective is to identify individual sound events along with their onset and end times.[1] A fundamental issue in the temporal segmentation of environmental sounds is what exactly constitutes an *event*. Even in a restricted class of sounds such as speech, any definition is somewhat ambiguous; "events" can refer to phonemes, words, sentences, or some other unit of organization. As such, we adhere as closely as possible to Bregman's concept of an *auditory stream*—which, according to the principle of *ecological validity*, is perceptually equivalent to the sound emanating from a single physical source [3]. A "speech event" would then consist of the continuous speech between pauses or transitions where the speaker changes—additional processing would be necessary to identify phonemes, words, or sentences. Likewise, a "typing event" would consist of clusters of keystrokes rather than a single keystroke.

Prior work in audio segmentation has generally focused on more limited domains, for instance speech [9] and music [10], [11]. In this paper, events are often defined as structurally meaningful units such as phonemes or musical notes. More recent approaches, however, do address the segmentation of continuously recorded environmental sound [4], [6], [12], [13]. Several methods [6], [12] address segmentation mainly in terms of high-level *semantic scenes* (e.g., "meeting," "supermarket," "library"). While these approaches are valuable in terms of indexing events contained within these scenes, they do not directly address the segmentation of individual sound events. In [4], the authors propose an event-based segmentation of continuous recordings made in an office environment. The method is based on heuristic thresholding of short-time Fourier transform (STFT) energy/spread features and hence may not scale to more dense environments. Our own past work [13] monitors rapid changes in six features (*loudness*, *spectral centroid*, *spectral sparsity*, *temporal sparsity*, *transient index*, *harmonicity*) that are specifically adapted to environmental and natural sounds, and which are described in Section II. This work incorporates a custom dynamic Bayesian network (DBN) that models how event boundaries influence changes in these features, allowing for uncertainties in terms of 1) which features change upon an event boundary, 2) how fast these features change, and 3) how each feature can vary during an event.

All of these event-based segmentation approaches are *single-channel*, using a single microphone placed somewhere in the

[1]If events overlap in time, physical source separation or "demixing" is also necessary; however, we neglect this issue at present. The source separation problem has been well studied (cf. [8]), and incorporating solutions into our proposed framework remains a topic of future work.

space. Many spaces such as office environments, however, are large enough that sounds originating in one part of the space may not be perceptible in another, or may be perceptible only at high SNR. To this end, [14], and [15] have used DBN approaches for event segmentation in office meetings with observations consisting of data collected from microphones and video cameras, but the locations of the audio sources were assumed known, either by attaching a lapel microphone to each speaker or steering a microphone array to a predefined direction. For continuously monitoring only sound environments, we extend our own previous work on DBN environmental sound segmentation [13] to the multichannel case, without making any assumptions of source type or location. Here, the sound is continuously recorded using a microphone array distributed throughout the space. Our method jointly infers onsets and end times of the most prominent sound events in the space, along with the *active subset* of microphones responsible for capturing the sound corresponding to each event. While not explicitly localizing the source, our method does yield a rough idea of where a certain sound event can be perceived within the space.

B. Example-Based Audio Retrieval

For the indexing and retrieval problems, the objective is that users should be able to retrieve all sound events of interest in an intuitive and efficient manner without too many false positives. By "intuitive," we mean that the user can quickly form the query; by "efficient," we mean that the system can quickly execute the retrieval. Due to a variety of both *recall-driven*[2] and *precision-driven* applications, we seek a flexible strategy that allows us to navigate the recall-precision tradeoff by varying retrieval size. A convenient strategy is *query-by-example* (QBE), where users input recordings they consider similar to the desired retrieval sounds. Users can either upload the query from a file or present it orally. Oral query is quite prevalent in melody retrieval in the form of query-by-humming (QBH) ([17]–[19], among others), where the sound objects to be retrieved concern melodies. During retrieval, the N database melodies considered "most similar" to the query are obtained.

As an *explicit model of query behavior*, we adopt a likelihood-based QBE strategy that computes the likelihood over all possible queries that arise as a result of each sound in the database. An optimal retrieval system will maximize the posterior $P(X \mid Y)$, where the query Y is the observation, and X is a hidden variable modeling the database sound that generated the query. With the additional assumption that all database sounds are equally likely (i.e., $P(X)$ is uniform), for a given query Y, we can rank all sounds in the database with respect to the likelihood $P(Y \mid X)$. As discussed in Section V, the observed query features Y are a *time series*, where each sample is a vector consisting of the six features discussed in Section II. The database feature set X also represents a time series in the same feature vectors; however, each individual feature trajectory is a hidden Markov model (HMM) that approximates a *general trend*, consisting of a zeroth-, first-, or second-order polynomial fit. These

[2]*Recall* (number of desired sounds retrieved/number of desired sounds in the database) and *precision* (number of desired sounds retrieved/number of retrieved sounds) are standard measures of retrieval performance; see [16] for further information.

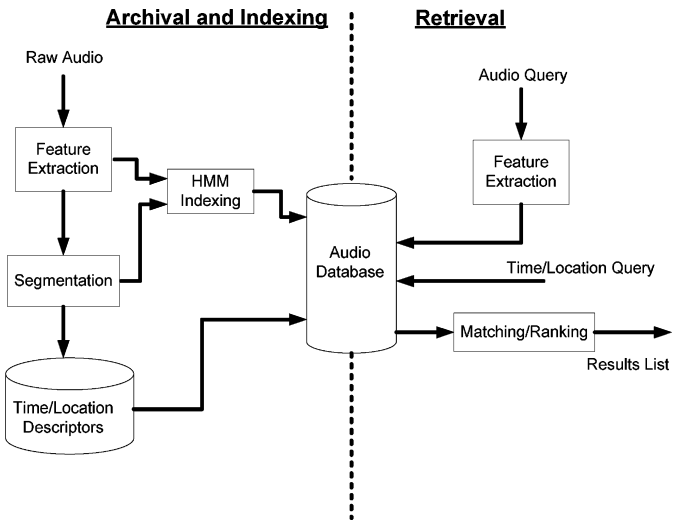

Fig. 1. Schematic diagram concerning the segmentation, indexing, and retrieval of continuously recorded environments.

fits suffice to encode whether the feature is constant (high or low), increasing/decreasing, or exhibiting more complex (up → down; down → up) behavior.

A drawback of likelihood-based QBE is that retrieval time is linear in the number of database sounds. Our target application, continuous monitoring, often involves large databases, which requires a faster approach. To this end, we have developed an approximate log-time retrieval method using recursive cluster-based indexing. Specifically, we present a novel distance semi-metric for calculating the similarity matrix between all archived sounds in the database, and a modified *spectral clustering* technique where the number of clusters are automatically determined using the Bayesian information criterion (BIC). Once clustering is complete, each cluster is fit with its own HMM template, and when a query is input into the system, only the cluster HMM which exhibits the highest likelihood has its sounds returned and ranked.

In remaining sections, we present specific details of our integrated approach concerning event-based segmentation, indexing, and example-based retrieval for environmental and natural sounds, which is diagrammed in Fig. 1. Section II describes the common feature set used in both segmentation and retrieval tasks. The probabilistic segmentation model for a single microphone is presented in Section III, while an extension to microphone arrays for monitoring large spaces is discussed in Section IV. Section V explains the query-by-example templates, which are used to retrieve segmented sound events from a database. An efficient approach to cluster-based indexing is examined in Section VI. The results of a formal user study are discussed in Section VII and used to evaluate the proposed retrieval system in terms of precision, recall, mean average precision, and computational cost. Finally, conclusions and suggestions for future work are provided in Section VIII.

II. Audio Feature Extraction

Efficient indexing and classification schemes for audio are typically based on the fundamental process of acoustic *feature extraction* [4], [6], [11], [20]–[22]. Indexing and retrieval

schemes based on these features are often referred to as *content-based*. Most acoustic feature extraction schemes rely on a frame-based analysis, where overlapping audio frames of 20–100 ms are windowed and used as the input for the feature extraction process. The reason for using frames of this length is, in general, dynamic audio sources tend to have some stationary characteristics, for instance spectral characteristics, at these time scales.

One of the first attempts at content-based indexing and retrieval of audio was the Muscle Fish database [20]. In this paper, a short-time analysis was performed on a database of approximately 400 sound files, employing four spectral features and the root mean square (rms) level of the sound. Statistics (mean, variance, and autocorrelation at small lags) of each feature trajectory were stored and used for the indexing and classification of all sounds in the database. Although the database of [20] contained a wide variety of sound types (speech, music, natural sounds, etc.) over half of their audio classes were musical instruments. In [21] four different feature sets (low-level signal parameters, MFCC, psychoacoustic, and auditory temporal envelope) are compared in terms of classification performance for general audio (speech, noise, crowds, etc.) and music genre classification. Although, the types of sounds in the general audio classes were far from complete, this paper presented for the first time the extremely important result that features that work well for music do not necessarily perform well on environmental and natural sounds.

A fundamental goal of the feature set in this paper is that features indicate the presence of a large variety of sounds while not specifically adapting to a particular type of sound, e.g., Mel frequency cepstral coefficients (MFCCs) for speech or chroma features for music. Additionally, we desire a relatively small, yet comprehensive feature set that can be efficiently calculated. Thus, we should choose only one from a set of functionally redundant features, for instance bandwidth and spectral sparsity, and avoid calculating an extremely large feature set at every frame, and then reducing the dimensionality using a method such as PCA. Furthermore, for indexing and retrieval we desire a feature set that provides intuitive meaning when searching for specific values of a given feature, something that we do not have for example with the fourth MFCC or third PCA coefficient. Although, other audio features, such as those in the MPEG-7 standard [23] might be similar to certain features in this work, we have found that the six features proposed in this section provide an intuitive and minimal set for both segmentation and retrieval purposes.

Due to the diversity of sound sources, we have found it necessary to calculate features at different time scales, from 40 ms (short-term) to one second (long-term). Short-term features are computed either directly from the windowed time series data or via STFT using overlapping 40-ms Hamming windows hopped every 20 ms. Long-term features are computed using a one-second sliding window to combine the data from 49 of the 40-ms windows, in order to capture slowly evolving textural characteristics of the sound. Using 98% overlap for the sliding window, (i.e., slide in 20-ms steps), both long and short-term features remain synchronous, and can be concatenated into a feature vector for each frame.

The first of our features is *loudness*, which we define as the rms level in decibels of the windowed frame indexed by t, i.e.,

$$Y_t^{(1)} \triangleq 20\log_{10}(\text{rms}_t). \tag{1}$$

Next, we compute from STFT data the Bark-weighted *spectral centroid*

$$Y_t^{(2)} \triangleq \frac{\sum_{j=1}^{M} b_j(b_j - b_{j-1})|X_t(j)|^2}{\sum_{j=1}^{M}(b_j - b_{j-1})|X_t(j)|^2} \tag{2}$$

where b_j is the frequency value of the center of STFT bin j in units of Barks [24]. A third feature, *spectral sparsity* is calculated from the zero-padded STFT data of each frame, via the ratio of ℓ^∞ and ℓ^1 norms calculated over the magnitude STFT spectrum (inspired by the common ℓ^1 sparsity metrics used for compression, blind source separation, etc. [25]). Defining $X_t(j)$ as the M-point STFT coefficient from frequency bin j for frame t, the spectral sparsity is defined as follows:

$$Y_t^{(3)} \triangleq \frac{\max(|X_t(1)|, \ldots, |X_t(M)|)}{\sum_{j=1}^{M}|X_t(j)|}. \tag{3}$$

Spectral sparsity should be large for pure sine tones or bells, and smaller for sounds with significant "noise" characteristics that imply a wide frequency spectrum.

As mentioned previously, our feature set also contains features computed using a 1-s sliding window that hops every 20 ms, constructed by combining data from $N = 49$ of the 40-ms frames. *Temporal sparsity* is one such feature, which we define as the ratio of ℓ^∞ and ℓ^1 norms calculated over the N small window rms values in a given one second window, i.e.,

$$Y_t^{(4)} \triangleq \frac{\max(\text{rms}_{t-(N+1)}, \ldots, \text{rms}_t)}{\sum_{k=t-(N+1)}^{t}\text{rms}_k}. \tag{4}$$

Temporal sparsity is large for sounds such as footsteps in relative silence and useful for indexing and retrieving these types of sounds.

Transient index is computed by averaging the cepstral flux from several frames of data. We define the transient index for frame t using the MFCCs [26] as follows:

$$Y_t^{(5)} \triangleq \sum_{k=t-(N+2)}^{t} \| \text{MFCC}_k - \text{MFCC}_{k-1} \|_2 \tag{5}$$

where MFCC_k is the 15th-order MFCC vector for frame k, and $N = 49$ signifies the number of short frames over which the transient index is computed. The transient index should be useful in detecting and segmenting sounds with spectral characteristics that exhibit consistent fluctuations between adjacent frames, e.g., crumpling newspaper.

Finally, *harmonicity* is used to measure probabilistically whether or not the STFT spectrum for a given frame exhibits a harmonic frequency structure. Harmonicity should be large for speech, music, and certain machine noises and smaller for most other types of environmental audio, as well as some

musical sounds (bells). Selecting the maximum peak in the autocorrelation function [23] is often used for harmonicity and fundamental frequency estimation. This approach is often biased towards peaks at very small lags, although this bias can be corrected with computationally expensive techniques based on a time-domain difference function [23]. We therefore choose a method based on partials in the frequency domain, which in our case is also computationally efficient since the STFT is already calculated for extraction of other features.

The algorithm begins by choosing the L most prominent frequency peaks from the STFT magnitude spectrum of frame t. Peak amplitudes are required to be above both a global and frame-adaptive threshold in order to provide robustness to noise. Then, the frequency of each peak in Hz, is stored in the set $\rho_t = \{f_1, \ldots, f_L\}$. Using an adaptation of Goldstein's algorithm [28], [29], we first estimate the fundamental frequency f_o, for which the kth harmonic has frequency kf_o. As discussed in [29], accurately computing the Goldstein pitch estimate requires solving a computationally intensive combinatoric optimization problem, and although highly accurate MCMC approximations have been proposed [29], [30], they are still too slow for our application. To provide an efficient approximation that works well in most real-world cases, we compute the Goldstein pitch estimate by searching pairwise combinations of the L frequency peaks, and $k_{\max} \leq 2L$ possible harmonics. Assuming the pair of peak frequencies denoted by $\{f_1, f_2\} \subset \rho_t$, are harmonics of f_o, with corresponding harmonic numbers of k_1 and k_2, the Goldstein likelihood [28], [29] can be computed from

$$P(f_1, f_2 \,|\, \hat{f}_o, k_1, k_2) = \prod_{j=1}^{2} \gamma(f_j; k_j f_o, \sigma_j) \tag{6}$$

where $\gamma(x; \mu, \sigma)$ is the univariate Gaussian pdf with mean μ and standard deviation σ evaluated at x, and the standard deviation terms σ_j are considered to be proportional to frequency [28], i.e.,

$$\sigma_j = C k_j f_o \tag{7}$$

with C a constant. In this paper, we set $C = 0.01/\sqrt{2}$, as this value was shown in [28] to provide accurate estimates for a wide range of harmonic numbers. Given the form of σ_j from (7), [28] shows that the fundamental frequency estimate $\hat{f}_o$ for any pair of frequency peaks is

$$\hat{f}_o = \frac{(f_1/k_1)^2 + (f_2/k_2)^2}{f_1/k_1 + f_2/k_2}. \tag{8}$$

The harmonicity $Y_t^{(6)}$ for frame t is then given by

$$\begin{aligned} &Y_t^{(6)} \triangleq \max_{f_1,f_2,k_1,k_2} P(f_1, f_2 \,|\, \hat{f}_o, k_1, k_2) \\ &\text{s.t.} : \{f_1, f_2\} \subset \rho_t, 1 \leq k_1 < k_2 \leq k_{\max}, \\ &f_1 < f_2, \text{and } \hat{f}_o > f_{\min}. \end{aligned} \tag{9}$$

The final constraint in (9) is used to avoid very low (1/2, 1/4, etc.) estimates for $\hat{f}_o$, while maximization of (9) is done by exhaustively evaluating (6) and (8) for all valid combinations of f_1, f_2, k_1, and k_2. If less than two frequency peaks are found in the magnitude spectrum for frame $t, Y_t^{(6)}$ is set to zero. This approach to harmonicity calculation treats the peak assignment as a nuisance parameter that is resolved using a "generalized" rather than a "marginalized" likelihood approach, where all frequency peaks are considered explicitly in the algorithm; however it is only the two most-likely peaks that are used in the pitch estimate.

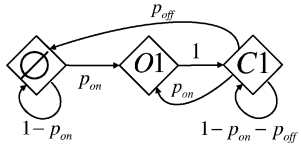

Fig. 2. Markov transition diagram for global mode M_t, where p_{new} is the prior probability of a new onset, while p_{off} is the prior probability of a sound turning off, given that it is currently on.

III. Feature-Based Audio Segmentation

Once the features from Section II have been extracted from the audio signal, we can use them to segment the recording into appropriate sound events. We employ a generative probabilistic model where event onsets and end points are inferred from the observed audio feature trajectories. Our model is implemented using a dynamic Bayesian network (DBN) framework which we now describe in detail.

Let $t \in 1:T$ be the frame index, and let K represent the number of features extracted from each frame of the signal, then $Y_t^{(i)}$, for $i \in 1:K$ are the observed audio features at time t. In our underlying model, we are not only concerned with separating regions of silence from regions where the sound is on, but we also hope to account for new sound clips beginning before the previous clip has turned off. This information is modeled by assigning a global *mode*, M_t to each audio frame of the recording. We represent M_t as a discrete random variable, which is responsible for controlling the underlying dynamics of all K features. The mode, M_t can take three possible values.

- $O1$-The onset, or beginning of a new sound event.
- $C1$-The continuation of a sound event between two successive frames.
- $\emptyset$ The absence of a perceptible sound event overlapping the given frame.

We assume M_t is a discrete first-order Markov process, governed by the Markov chain shown in Fig. 2. When a sound event begins (transitions from $M_{t-1} = \emptyset$ to $M_t = O1$ or $M_{t-1} = C1$ to $M_t = O1$) or ends (transition from $M_{t-1} = C1$ to $M_t = \emptyset$) we expect to observe large changes in the observed feature values at these frames. Additionally, by assigning a nonzero probability to transitions from $M_{t-1} = C1$ to $M_t = O1$, our model explicitly accounts for overlapping sound events.

Because of the variation in time scales and meaning of the different features, it is possible that certain features lag behind the overall mode M_t when turning on or off. Furthermore, even if there is a sound present at time t, it is likely that some of the audio features will fail to respond at all. The discrete random variables $\mu_t^{(i)}$, for $i \in 1:K$ serve as gates modeling the responsiveness and delays between the onset and end times of the different features. Similar to $M_t, \mu_t^{(i)} \in \{\emptyset, O1, C1\}$, where the definitions of $\emptyset, O1$, and $C1$ are the same as for M_t.

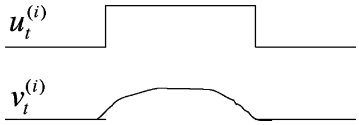

Fig. 3. Inherent feature $v_t^{(i)}$, and auxiliary variable $u_t^{(i)}$, which are the components of the hidden state vector $S_t^{(i)}$ for feature i.

Because event onsets are not instantaneous, but rather fade in/out over multiple frames, we also define the hidden *states* $S_t^{(i)}$ for $i \in 1:K$, which are continuous random variables mediating the effect of individual features' onsets/end times on the actual feature observations $Y_t^{(i)}$. The latter are modeled as *inherent* features corrupted by noise. As shown in Fig. 3, the state $S_t^{(i)} = [u_t^{(i)}, v_t^{(i)}]^T$ is then composed of this inherent feature plus an auxiliary variable enabling $S_t^{(i)}$ to be encoded as a first order Gauss–Markov process in time.

Assuming T available data frames and K features, the sequence of observations can be written as $Y_{1:T}^{(1:K)}$, meaning a sequence of T observations for each of the K features. Similar notation is used to represent the sequences of hidden variables, $S_{1:T}^{(1:K)}$ and $\mu_{1:T}^{(1:K)}$, while all features share a common global mode sequence, $M_{1:T}$. The directed acyclic graph (DAG) for our proposed feature fusion segmentation model with K features is shown in Fig. 4, where the presence of a directed arc between two nodes indicates dependence between the random variables represented by those nodes, while the lack of an arc between two nodes represents independence. Thus, Fig. 4 specifies the factorization of the joint distribution $P(M_{1:T}, \mu_{1:T}^{(1:K)}, S_{1:T}^{(1:K)}, Y_{1:T}^{(1:K)})$, where all K features are assumed independent.

A. Distributional Specifications

To define the segmentation model of Fig. 4, we must specify conditional probability distributions $P(Y_t^{(i)} \mid S_t^{(i)}), P(S_t^{(i)} \mid S_{t-1}^{(i)}, \mu_t^{(i)}, \mu_{t-1}^{(i)}), P(\mu_t^{(i)} \mid \mu_{t-1}^{(i)}, M_t,)$ M_{t-1}, and $P(M_{t+1} \mid M_t)$. We begin by modeling the likelihood of the observed features $Y_t^{(i)}$ as inherent features corrupted by Gaussian noise

$$P\left(Y_t^{(i)} \mid S_t^{(i)}\right) = \mathcal{N}\left(v_t^{(i)}, R^{(i)}\right) \tag{10}$$

where $R^{(i)}$ is the observation noise variance for feature i, and $v_t^{(i)}$ is the inherent feature (Fig. 3) and one component of the continuous random state vector $S_t^{(i)}$. The continuous state variables $u_t^{(i)}$ and $v_t^{(i)}$ satisfy the following stochastic recursive relations

$$\begin{aligned} u_t^{(i)} &= u_{t-1}^{(i)} + q_t^{(i)}\left(\mu_t^{(i)}, \mu_{t-1}^{(i)}\right) \\ v_t^{(i)} &= \left(1-\alpha^{(i)}\right) u_t + \alpha^{(i)} v_{t-1}^{(i)} \end{aligned} \tag{11}$$

where $\alpha^{(i)}$ is a low-pass filter coefficient, which allows for rapid but non-instantaneous change of the inherent feature across segments. Scalar process noise $q_t^{(i)}(\mu_t^{(i)}, \mu_{t-1}^{(i)})$ is distributed according to

$$q_t^{(i)}\left(\mu_t^{(i)}, \mu_{t-1}^{(i)}\right) \sim \mathcal{N}(0, Q^{(i)}\left(\mu_t^{(i)}, \mu_{t-1}^{(i)}\right)). \tag{12}$$

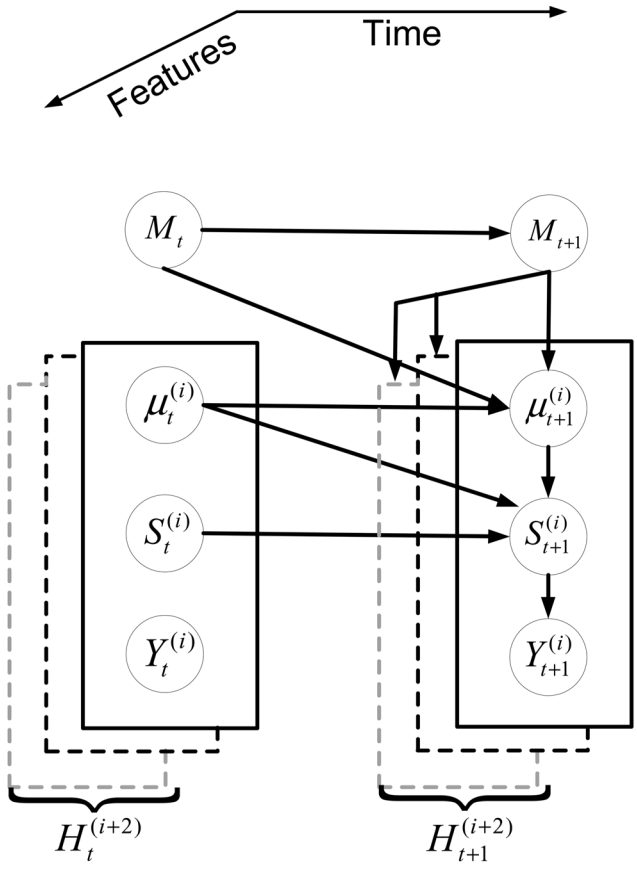

Fig. 4. Directed acyclic graph for feature fusion audio segmentation model, where $H_t^{(i)}$ denotes all variables for feature i.

where $Q^{(i)}(\mu_t^{(i)}, \mu_{t-1}^{(i)})$ is a variance that will be large during event onset and end times ($\mu_t^{(i)} \neq \mu_{t-1}^{(i)}$) and small when changes do not occur ($\mu_t^{(i)} = \mu_{t-1}^{(i)}$). In other words, the observed feature value at time t will be close to the value at $t-1$, when $\mu_t^{(i)} = \mu_{t-1}^{(i)}$, and far from the value at $t-1$ when $\mu_t^{(i)} \neq \mu_{t-1}^{(i)}$. See [31] for further discussion on choosing this variance value. It is also possible to describe the dynamics of the state vector $S_t^{(i)}$ defined in (11), using a state-space equation

$$S_t^{(i)} = A^{(i)} S_{t-1}^{(i)} + B^{(i)} q_t^{(i)}\left(\mu_t^{(i)}, \mu_{t-1}^{(i)}\right) \tag{13}$$

where

$$A^{(i)} = \begin{bmatrix} 1 & 0 \\ 1-\alpha^{(i)} & \alpha^{(i)} \end{bmatrix}, \quad B^{(i)} = \begin{bmatrix} 1 \\ 1-\alpha^{(i)} \end{bmatrix}. \tag{14}$$

Using (13) we now specify the conditional distribution of $P(S_t^{(i)} \mid S_{t-1}^{(i)}, \mu_t^{(i)}, \mu_{t-1}^{(i)})$ as

$$\begin{aligned} &P\left(S_t^{(i)} \,\middle|\, S_{t-1}^{(i)}, \mu_t^{(i)}, \mu_{t-1}^{(i)}\right) \\ &\quad = \mathcal{N}\left(A^{(i)} S_{t-1}^{(i)}, B^{(i)} Q^{(i)}\left(\mu_t^{(i)}, \mu_{t-1}^{(i)}\right)\left(B^{(i)}\right)^T\right). \end{aligned} \tag{15}$$

Via $P(\mu_t^{(i)} \mid \mu_{t-1}^{(i)}, M_t, M_{t-1})$, we model possible lags between when a particular gate $\mu_t^{(i)}$ turns on after M_t has turned on as Poisson. Letting $p_{\text{lag+}}^{(i)}$ be the probability that the lag will continue for an additional frame, the expected lag becomes $1/p_{\text{lag+}}^{(i)}$.

TABLE I
MODE TRANSITION PROBABILITIES FOR $P(\mu_t^{(i)} \,|\, \mu_{t-1}^{(i)}, M_t, M_{t-1})$

M_t	M_{t+1}	$\mu_t^{(i)}$	$P(\mu_{t+1}^{(i)} = \emptyset)$	$P(\mu_{t+1}^{(i)} = O1)$	$P(\mu_{t+1}^{(i)} = C1)$
$\emptyset$	$\emptyset$	$\emptyset$	1	0	0
$\emptyset/O1/C1$	$\emptyset$	$O1/C1$	$1 - p_{\text{lag}-}^{(i)}$	0	$p_{\text{lag}-}^{(i)}$
$\emptyset/O1/C1$	$O1/C1$	$\emptyset$	$1 - p_{\text{lag}+}^{(i)}$	$p_{\text{lag}+}^{(i)}$	0
$\emptyset/C1$	$O1$	$O1/C1$	$p_{\text{lag}+}^{(i)} - (p_{\text{lag}-}^{(i)} \cdot p_{\text{lag}+}^{(i)})$	$1 - p_{\text{lag}+}^{(i)}$	$p_{\text{lag}-}^{(i)} \cdot p_{\text{lag}+}^{(i)}$
$C1$	$C1$	$O1/C1$	0	0	1
$O1$	$C1$	$O1$	0	0	1
$O1$	$C1$	$C1$	$p_{\text{lag}+}^{(i)} - (p_{\text{lag}-}^{(i)} \cdot p_{\text{lag}+}^{(i)})$	$1 - p_{\text{lag}+}^{(i)}$	$p_{\text{lag}-}^{(i)} \cdot p_{\text{lag}+}^{(i)}$

Similarly, we model possible lags between when a particular gate $\mu_t^{(i)}$ turns off after M_t has turned off as Poisson, with $p_{\text{lag}-}^{(i)}$ as the probability that the lag will continue for an additional frame. A summary of $P(\mu_t^{(i)} \,|\, \mu_{t-1}^{(i)}, M_t, M_{t-1})$ for all possible combinations of $\mu_{t-1}^{(i)}, M_t$, and M_{t-1}, is shown in Table I. For each feature, we must specify a value for $p_{\text{lag}+}^{(i)}$ and $p_{\text{lag}-}^{(i)}$. As an example, we have observed that features based on spectral peaks (e.g., harmonicity, spectral sparsity) tend to indicate event end times before features such as spectral centroid by approximately 20 frames, thus, a value of $p_{\text{lag}-}^{(i)} = 0.05$ would be appropriate for spectral centroid.

Finally, we specify $P(M_{t+1} \,|\, M_t)$ as the Markov chain in Fig. 2, which requires values for p_{new} and p_{off}, the prior probabilities of a sound turning on and off, respectively. Setting these priors to $p_{\text{new}} = 0.001$ $p_{\text{off}} = 0.002$ roughly corresponds to incorporating into the model the prior information that a sound event of length 500 frames occurs once every 1000 frames. In general we have found that the segmentation model is relatively robust to incorrect initial settings of the prior probabilities ($p_{\text{new}}, p_{\text{off}}, p_{\text{lag}+}^{(i)}$, etc.) within approximately two orders of magnitude.

B. Inference Methodology

Segmentation is achieved by estimating the global mode sequence $M_{1:T}$. Ideally, our estimation criterion should preserve the correct number of segments, and the detected segment boundaries should be near the true segment locations. In order to achieve these goals we choose the maximum *a posteriori* (MAP) or global segmentation criterion to estimate $M_{1:T}$. The MAP criterion can be defined as

$$\hat{M}_{1:T} = \arg\max_{M_{1:T}} P\left(M_{1:T} \,\middle|\, Y_{1:T}^{(1:K)}\right) \tag{16}$$

and is typically more effective in estimating segmentation sequences that do not contain boundaries in adjacent frames, in contrast to a local frame error segmentation criterion [11].

Unfortunately, computing the exact MAP estimate requires exponential-time complexity. A linear-time approximation nevertheless exists, using an approximate Viterbi inference scheme [32]. To use this scheme for our segmentation problem, we begin by combining the $K+1$ discrete nodes M_t and $\mu_t^{(1:K)}$ from the DAG in Fig. 4 into a single discrete random variable $\Psi_t = M_t \times \mu_t^{(1)} \ldots \times \mu_t^{(K)}$, where $\times$ represents the Cartesian product and Ψ_t can take $|\Psi| = 3^{K+1}$ possible values. We then write the posterior of the sequence $\Psi_{1:T}$ as

$$P\left(\Psi_{1:T} \,\middle|\, Y_{1:T}^{(1:K)}\right) \propto P(\Psi_1) \prod_{t=2}^{T} P(\Psi_t \,|\, \Psi_{t-1}) \prod_{i=1}^{K} \left[P\left(Y_1^{(i)} \,|\, \Psi_1\right) \prod_{t=2}^{T} P\left(Y_t^{(i)} \,\middle|\, \Psi_{1:t}, Y_{1:t-1}^{(i)}\right) \right] \tag{17}$$

where the last term in (17) is the observation likelihood of the Kalman filter [31], [32] and

$$P(\Psi_t \,|\, \Psi_{t-1}) = P(M_t \,|\, M_{t-1}) \times \prod_{i=1}^{K} P\left(\mu_t^{(i)} \,\middle|\, \mu_{t-1}^{(i)}, M_t, M_{t-1}\right). \tag{18}$$

The Viterbi recursions can then be summarized by

$$P\left(Y_{t+1}^{(1:K)} \,\middle|\, \Psi_{1:t+1}, Y_{1:t}^{(1:K)}\right) \approx \prod_{i=1}^{K} P\left(Y_{t+1}^{(i)} \,\middle|\, \Psi_{1:t-1}^{*}(\Psi_t), \Psi_t, \Psi_{t+1}, Y_{1:t}^{(i)}\right) \tag{19}$$

where

$$\Psi_{1:t-1}^{*}(\Psi_t) \approx \arg\max_{\Psi_{1:t-1}} P\left(\Psi_{1:t-1} \,\middle|\, \Psi_t, Y_{1:t}^{(1:K)}\right). \tag{20}$$

The complexity of the approximate Viterbi inference algorithm is $\mathcal{O}(|\Psi|^2 T)$, thus, complexity is exponential in the number of features used for segmentation. For this reason, it is desirable to use as few features as possible for segmentation, while all six features are used for indexing and retrieval. In the results presented in Sections III-C and Sections IV-B, three features are used for segmentation.

Additionally, approximate Viterbi inference can be combined with an expectation maximization (EM) algorithm for estimating the noise variances and transition probabilities of Section III-A, cf. [32]. Although, since the EM algorithm is not convex, to guarantee convergence to an acceptable solution the model parameters must initially be chosen using knowledge of the audio data and physical meaning of the parameters as discussed in Section III-A.

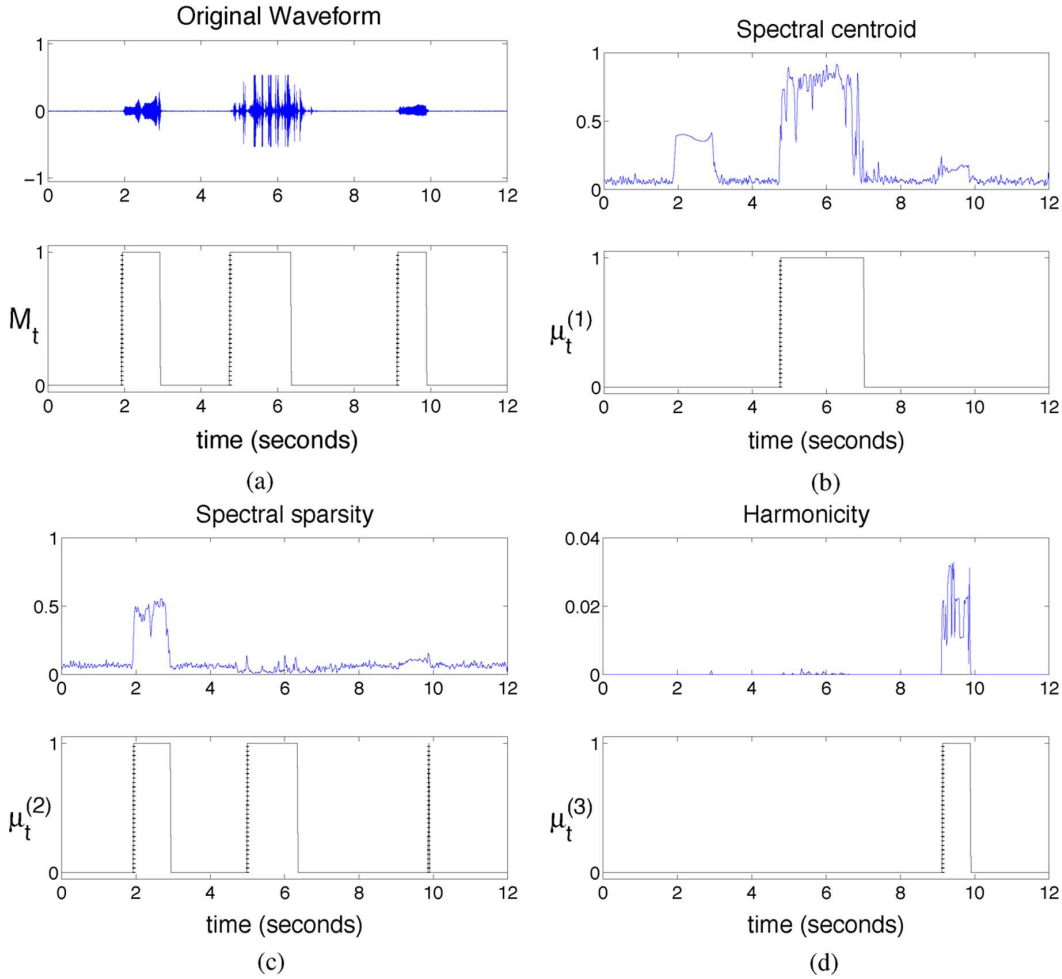

Fig. 5. Spectral feature segmentation example in an indoor environment. (a) Signal waveform (upper panel) and global mode $M_{1:T}$ (lower panel). (b)–(d) Extracted audio features (upper panel) along with fusion gating parameter, $\mu_{1:T}^{(1;3)}$ (lower panel).

C. Single-Channel Segmentation Results

To illustrate the performance of the feature extraction and segmentation algorithms discussed in this paper, we apply them to example audio recordings in both indoor and outdoor environments. All sound files were captured and processed at 16 bits/44.1 kHz uncompressed. Prior to segmentation, all extracted features were normalized to [0, 1], and the approximate Viterbi inference scheme was used to infer the segmentation.

Fig. 5 displays results for an indoor recording of a whistle (2–3 s), a key jingle (4.5–7 s), and a shout (9–10 s). The time-domain waveform for this sound file is shown in the upper plot of Fig. 5(a), while the extracted global mode sequence $M_{1:T}$ is displayed directly beneath the time domain waveform. Mode values of $M_t = \emptyset$ are plotted as zero, $M_t = C1$ are plotted as one, and $M_t = O1$ are represented by dashed lines. The top plots of Fig. 5(b)–(d) displays the audio feature sequences corresponding to spectral centroid, spectral sparsity, and harmonicity, respectively. The bottom plots of Fig. 5(b)–(d) contains the inferred feature fusion parameters $\mu_{1:T}^{(i)}$ for $\mathrm{i} = 1, 2, 3$, respectively.

The example from Fig. 5 helps to illustrate the diversity of our chosen feature set as each of the three spectral features indicate strongest for different events. The spectral centroid [Fig. 5(b)] detects the key jingle between 4.5 and 7 s, while only increasing slightly during the whistle and shout sounds. As expected, spectral sparsity [Fig. 5(c)] is highest during the non-harmonic whistle sound, while only slightly indicating the presence of the key jingle and shout. Harmonicity [Fig. 5(d)] only contributes to the segmentation and indicates the presence of a sound event during the shout sound, which as it is generated by the human voice tends to exhibit harmonic structure.

A second example demonstrating the performance of the segmentation algorithm is shown in Fig. 6, which was recorded outdoors in particularly windy conditions. The sound is footsteps on metal bleachers (14–19 s). Examining the time domain waveform shown in the top panel of Fig. 6(a), we see that there are no clearly visible sound events, although the global mode sequence (segmentation) shown in the bottom panel does extract the footsteps even in these low SNR conditions. Due to the low SNR, loudness is of no use for segmentation as shown in Fig. 6(b). Fortunately, the spectral centroid and spectral sparsity features shown in Fig. 6(c) and (d), respectively, are able to detect the presence of the footsteps. This example helps to demonstrate the performance of the algorithm in low SNR conditions.

Next, we examine the performance of the proposed segmentation algorithm on approximately two hours of continuous

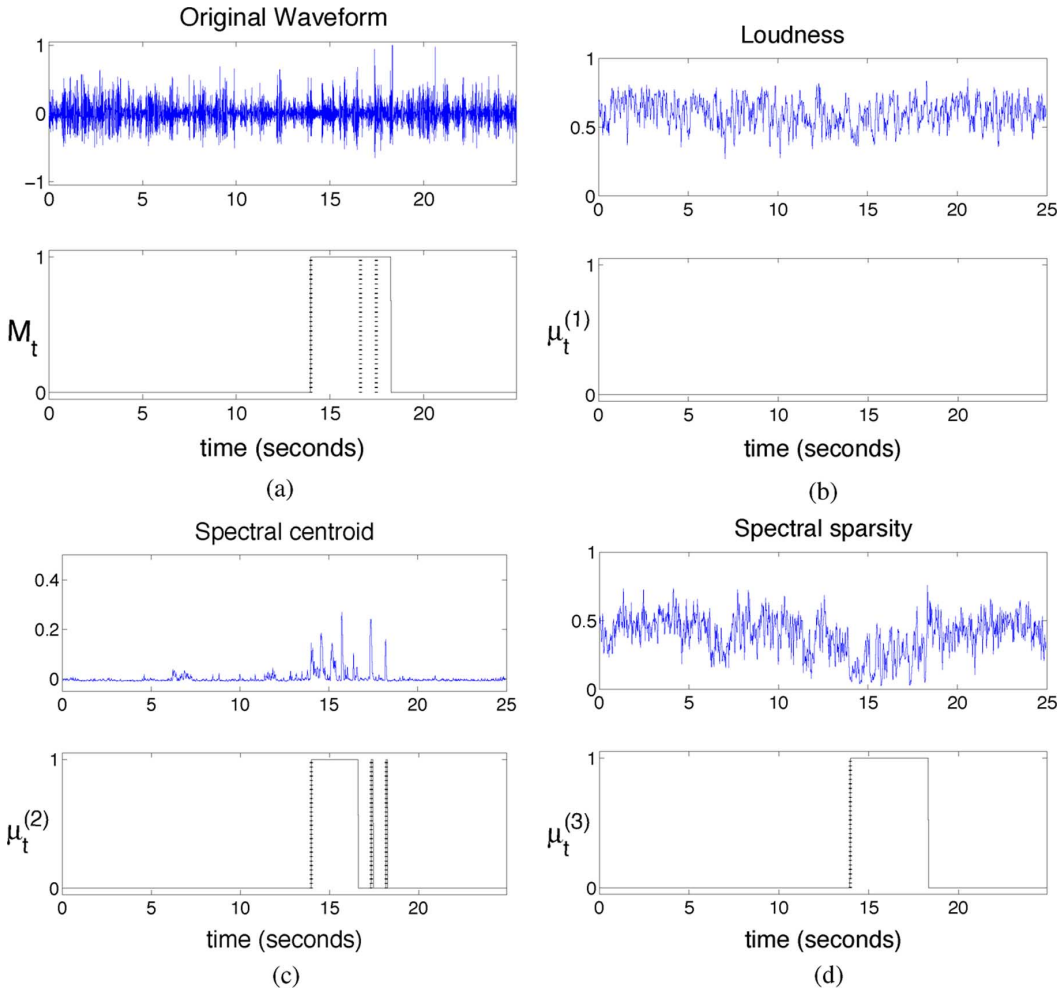

Fig. 6. Multi-feature segmentation example for footsteps on metal sound recorded in a windy outdoor environment. (a) Signal waveform (upper panel) and global mode $M_{1:T}$ (lower panel). (b)–(d) Extracted audio features (upper panel) along with fusion gating parameter $\mu_{1:T}^{(1:3)}$ (lower panel).

recordings. Approximately one hour was recorded outdoors in a park during a windy day and contains sounds of people playing sports, talking, and footsteps amongst other sounds. The rest of the data was recorded indoors in an apartment and contains sounds related to telephone conversations, cleaning, and cooking among others. The recordings were manually segmented into 155 indoor events and 170 outdoor events, and we tested the performance of our proposed segmentation algorithm using this manual segmentation as ground truth. Additionally, we used the BIC segmentation algorithm originally proposed for speaker segmentation [33], and used for sound environment segmentation in [34] as a baseline system. Theoretically, the penalty term in the BIC segmentation algorithm should be one, but we found that this was far too sensitive for our application, so we treated the penalty term as a tunable parameter [34].

Table II displays the recall and precision of the proposed segmentation approach and the modified BIC approach with tunable penalty parameter. For both approaches, the loudness, spectral centroid, and spectral sparsity features were used as the inputs to the algorithm. Approximately one minute of data was used to tune the prior and noise variance parameters in the proposed DBN approach and the penalty parameter in the modified BIC approach. For segmentation, precision is the number of correctly identified event onsets divided by the total number of event onsets returned by the algorithm (similar to the inverse of false alarm rate), and recall is the number of correctly identified event onsets divided by the true number of event onsets (analogous to the detection rate). We see from Table II that the proposed DBN approach outperforms the modified BIC approach, with the most notable improvements for the outdoor recordings. In this case, the loudness feature is not useful for segmentation (due to a high amount of background and wind noise), which seems to negatively influence the BIC approach, while our approach was designed specifically to perform well in situations where certain features do not always indicate the presence of an event. Furthermore, the proposed approach is based on approximate Viterbi inference, which is a linear time algorithm, while the BIC segmentation algorithm has quadratic time complexity. One downside of the proposed approach is that there are more parameters to tune/learn (the prior for the global mode as well as the gate prior and noise variances for each feature), while the BIC approach has a single adjustable parameter.

IV. Multi-Channel Segmentation

In most real-world situations, a single microphone is insufficient for the characterization of an entire auditory scene. These

TABLE II
SEGMENTATION PERFORMANCE FOR APPROXIMATELY 2 h OF CONTINUOUS RECORDINGS IN BOTH INDOOR AND OUTDOOR ENVIRONMENTS

Segmentation Algorithm	Indoor		Outdoor	
	Precision	Recall	Precision	Recall
Proposed DBN	0.560	0.664	0.5067	0.665
Modified BIC	0.467	0.607	0.3729	0.3860

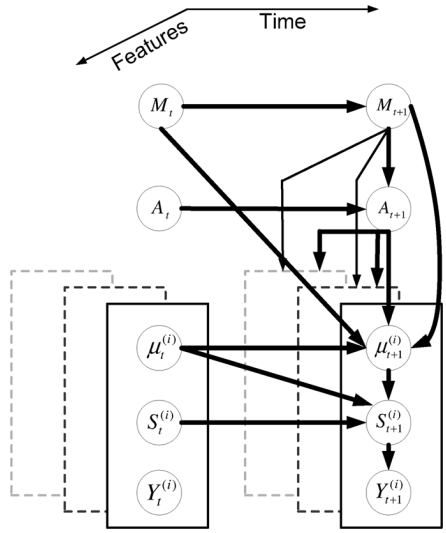

Fig. 7. Directed acyclic graph for multi-channel audio segmentation model.

scenes often occur in places such as office environments which consist of several interconnected spaces that are at least partially acoustically isolated from one another. To this end, we perform the scene characterization using a sparse array of microphones, strategically placed to ensure that all parts of the environment are within range of at least one microphone. By adding an additional layer to the segmentation model of Section III that accounts for which microphones are *active* for a given sound event, we can perform a *multi-channel* segmentation that can capture sound events occurring in any part of the space and also increase noise robustness [35]. Furthermore, by explicitly inferring the active subset of microphones corresponding to a given sound event, we can establish a rough estimate of where the event occurs without resorting to explicit beamforming/localization techniques that may be inaccurate at low SNR or impossible if microphones are located several meters from one another.

Similar to the single feature segmentation model described in Section III, the multi-channel model begins with the global mode, M_t, which is shared by all features and microphones. The three values M_t can take are $\emptyset, O1, C1$ as described in Section III. We assume microphone locations are known, but we have no knowledge of the number, type, and location of the sound sources we want to segment. Thus, we consider an event onset/end time as a frame where large "jumps" are observed in the trajectories of one or more features at one or more of the microphones. Letting N denote the number of microphones, we define the time-varying N-dimensional *active subset* vector A_t, for which elements $A_{t,n} \in \{0,1\}$ for $n = 1, \ldots, N$ indicate whether the nth microphone is active at frame t. The active subset A_t can take 2^N possible values, one for each possible combination of active/inactive microphones. If there are acoustically isolated microphones, and several acoustically isolated events occur simultaneously, only the sound with the highest SNR will be detected. This is because it is highly unlikely that spatially isolated microphones will be included in the same active subset, and there can only be a single active subset for each time frame.

The multi-channel segmentation model also contains an individual feature gating variable $\mu_t^{(i)}$, for $i \in 1:K$, where $\mu_t^{(i)}$ is now an N-dimensional vector constrained by the active subset. For instance, suppose $N = 4$ and $A_t = [0,1,0,1]^T$; then $\mu_t^{(i)} \in \{[\emptyset,\emptyset,\emptyset,\emptyset]^T, [\emptyset, O1, \emptyset, O1]^T, [\emptyset, C1, \emptyset, C1]^T\}$. That is, only active microphones can have features that behave differently from silence.[3] Furthermore, by treating A_t and $\mu_t^{(i)}$ as vector-valued random variables, contributions from all microphones are used in determining the final segmentation.

The observed features $Y_t^{(i)}$ are also N-dimensional vectors in the multi-channel case, where the observation of feature i at frame t for the nth microphone is denoted by $Y_{t,n}^{(i)}$. The observations between all K features, and N microphones are assumed to be independent, and similar to the single feature case each observation $Y_{t,n}^{(i)}$ has a hidden state $S_{t,n}^{(i)}$ associated with it, composed of an inherent feature and auxiliary variable as described in Section III.

Assuming T frames, K audio features, and N microphones, the multi-channel feature fusion segmentation model leads to the DAG of Fig. 7 for the joint distribution $P(M_{1:T}, A_{1:T}, \mu_{1:T}^{(1:K)}, S_{1:T}^{(1:K)}, Y_{1:T}^{(1:K)})$.

A. Distributional Specifications

Many of the conditional probability distributions necessary to fully describe the distribution of Fig. 7 are identical to those described in Section III-A for the single-channel case, only now there are N independent observations for each feature instead of one. Although there are almost surely feature correlations between the N different microphones, these depend on a variety of unknown factors including the sound source and operating environment. Thus, we assume independence rather than trying to model dependencies based on knowledge we do not possess. We now describe the important differences between the single and multi-channel segmentation models.

The feature gate variables $\mu_{t,n}^{(i)}$ model whether or not feature i is active for microphone n, using the vector distribution $P(\mu_t^{(i)}|\mu_{t-1}^{(i)}, M_t, M_{t-1}, A_t)$. In general, this conditional distribution is the same as that described for the single feature case, only with dependence on A_t. The dependence on A_t allows only microphones included in the active subset to have their gate $\mu_{t,n}^{(i)}$ behave differently from silence, while all microphones included in the active subset must share the same value for $\mu_{t,n}^{(i)}$.

The active subset variables A_t, which index the microphones that observe specific sound events are described by the distribution $P(A_{t+1}|A_t, M_{t+1})$. The distribution $P(A_{t+1}|A_t, M_{t+1})$

[3]If microphones are located very far from each other, while an extremely loud sound source is located near a subset of the microphones, it is possible that a sound arrives at some microphones one or more frames before it arrives at others. Due to the large inter-microphone distances, it is unlikely that these microphones will be included in the same active subset, thus all active microphones should share the same value of the feature gating variable.

behaves in three distinctly different ways depending on the value of the global mode M_{t+1}. When there are no sound events observed by any of the microphones ($M_{t+1} = \emptyset$) then the active subset will be empty with probability one, i.e., $P(A_{t+1} = \emptyset | A_t, M_{t+1} = \emptyset) = 1$. When a sound event continues between consecutive frames ($M_{t+1} = CI$), then the active subset is constrained to be $P(A_{t+1} = A_t | A_t, M_{t+1} = CI) = 1$, i.e., the active subset must be constant over an entire sound event. Thus, the active subset only changes during the onset of new sound events ($M_{t+1} = OI$).

If a sound event has an onset at time $t+1$, the active subset at time $t+1$ is independent of the active subset at time t, i.e., $P(A_{t+1}|A_t, M_{t+1} = OI) = P(A_{t+1}|M_{t+1} = OI) = P(U)$, where $U \in \{1, 2, \ldots, 2^{N-1}\}$ are all the non-empty active subset possibilities. The formulation for $P(U)$ should encode the prior knowledge that subsets of closely spaced microphones should all observe the same sound. To determine $P(U)$ we first define one active microphone as the anchor or reference microphone denoted by $\gamma \in \{1, \ldots, N\}$ for each non-empty active subset. Each microphone is considered as a possible anchor, and we marginalize the likelihood over all possibilities, i.e.,

$$P(U) = \sum_{\gamma} P(U|\gamma) P(\gamma) \tag{21}$$

where the anchor microphone is uniformly distributed, i.e., $P(\gamma) = 1/N$. The probability of each possible active subset given an anchor microphone is

$$P(U|\gamma) = \prod_{A_{t,n}=1} p_{\text{on}}^{(n)} \prod_{A_{t,n}=0} (1 - p_{\text{on}}^{(n)}) \tag{22}$$

where $A_{t,n} = 1$ signifies that the nth microphone is active at time t, and $A_{t,n} = 0$ signifies inactivity. The probability that the nth microphone is active $p_{\text{on}}^{(n)}$ is given by an inverse sigmoidal function

$$p_{\text{on}}^{(n)} = \frac{1}{1 + \exp(a\varepsilon^{(\gamma,n)} - b)} \tag{23}$$

where a and b are parameters controlling the slope and offset of the sigmoid function, and $\varepsilon^{(\gamma,n)}$ is the Euclidean distance between microphone n and the anchor microphone γ. Thus, by choosing the inverse sigmoid model of (23) for $p_{\text{on}}^{(n)}$ only those microphones located close to the anchor microphone will have a high probability of observing the same sound event as the anchor microphone.

Inference in the multi-channel segmentation algorithm is also performed using the Viterbi scheme described in Section III-B; however, the number of discrete nodes is increased to $|\Psi| = 2^N 3^{K+1}$ due to the active subset layer in the model of Fig. 7. For the five microphone ($N = 5$) and three feature ($K = 3$) segmentation experiments described below there are $|\Psi| = 2592$ possible discrete states, and the order of complexity is $\mathcal{O}(|\Psi|^2 T)$, which makes adding additional microphones or acoustic features difficult when using the Viterbi approximation. In order to make inference tractable, a Monte-Carlo inference scheme, e.g., the Rao–Blackwellised particle filter (RBPF) [36] can be used when a very large microphone array is required, and evaluation of these approximate inference schemes remains a topic of future work.

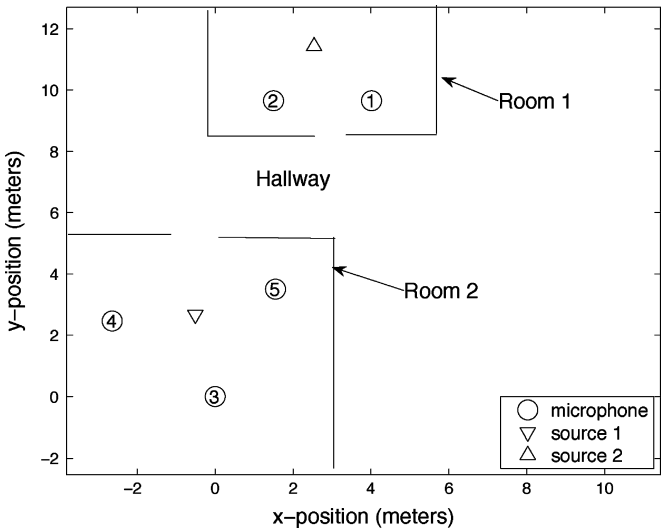

Fig. 8. Floor plan of office environment, microphone array positions, and approximate locations of two sound sources.

B. Multi-Channel Segmentation Results

The multi-channel segmentation algorithm is examined in a similar manner to the single microphone case described in Section III-C. Our example recordings were generated using a five microphone array of an indoor office environment consisting of two meeting rooms separated by a hallway. All microphones in the array had equal elevation, and (x, y) coordinates in meters of (4.01, 9.65), (1.50, 9.65), (0, 0), (−2.64, 2.46), and (1.55, 3.51), where (0, 0) was an arbitrarily chosen reference point selected to be the location of the third microphone. Fig. 8 shows the floor plan of the office environment, the location of the microphones, and the approximate positions of two example sound sources.

Fig. 9(a) displays the time-domain waveforms from each of the five microphones in the array for an example recording. The global mode sequence $M_{1:T}$ inferred from the Viterbi algorithm is shown in the bottom panel of Fig. 9(a), while the inferred active subset variables $(A_{t,n})$ are plotted below the corresponding channel waveforms. Values when the global mode is off ($M_t = \emptyset$) are plotted as zero, values when the global mode is on ($M_t = C1$) are plotted as one, and onsets ($M_t = O1$) are plotted as dotted lines. From Fig. 9(a) two distinct events from this example recording can be recognized. First, there is the sound of jingling keys (2–4 s) on channels three, four, and five. Second, there is another key jingle sound (3–6 s) on channels one and two. As a baseline system, Fig. 9(b) displays the "downmixed" signal where all five channels are averaged together and the single-channel segmentation algorithm described in Section III is used to segment the signal. The loudness, spectral centroid, and spectral sparsity features are used as the input to both the single and the multi-channel algorithms.

From Fig. 9(b) we see that the single-channel baseline system cannot distinguish between the two overlapping key

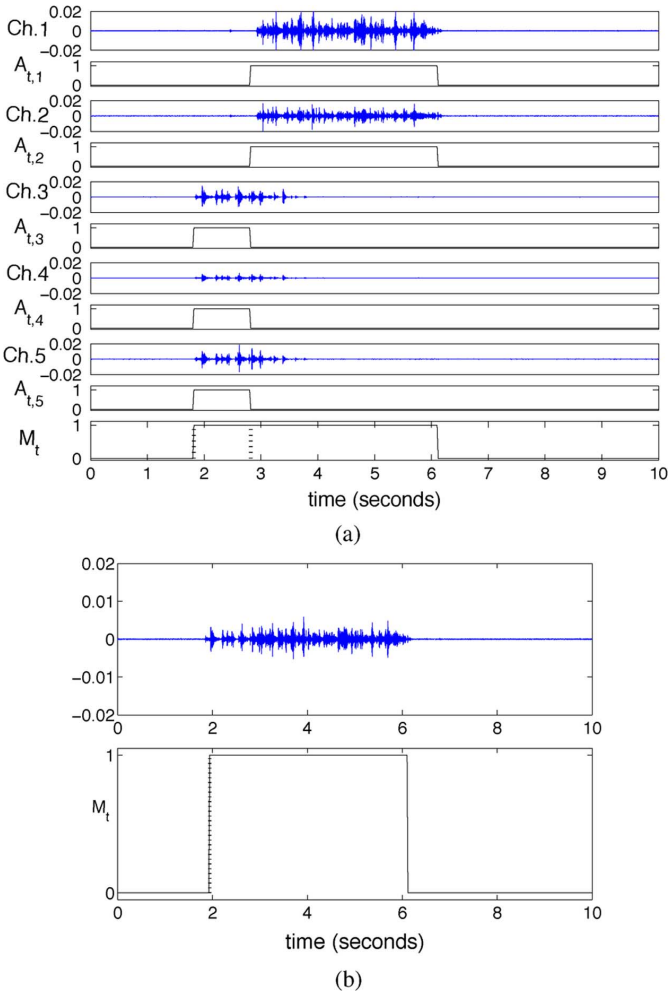

Fig. 9. Multi-channel segmentation example. (a) Signal waveforms for five microphone indoor recording along with the corresponding active subset variables $A_{t,n}$, and the global mode sequence $M_{1:T}$. (b) The downmixed signal of the five channels and the global mode sequence $M_{1:T}$ extracted using the single-channel segmentation algorithm.

jingle events. As shown in Fig. 9(a) the multi-channel algorithm is able to detect the point when the second event begins, but during the time when the two sounds overlap only the highest energy event (the one occurring on channels one and two) is detected. After parameter tuning, the downmixed single-channel and multi-channel segmentation algorithms were also compared on a collection of 19 additional sound events recorded using the array of Fig. 8, and concatenated into a continuous recording. This collection contained both overlapping and non-overlapping events such as speech, laughter, key jingles, and shaking containers. The behavior shown in the example of Fig. 9 was typical as the downmixed single-channel algorithm missed ten segments either due to overlapping events of the same type blending together after downmixing, or noise from the downmixing process drowning out low energy sounds. The multi-channel algorithm missed three segments due mainly to situations where events overlapped in time on different microphones, but only the highest energy event was detected. In terms of false alarms, the performance of both algorithms was similar, as there were six extra onsets for the single-channel segmentation algorithm, and seven extra onsets for the multi-channel algorithm.

V. Query-by-Example for Efficient Retrieval

Our QBE method is based on retrieving the R database sounds with the greatest likelihoods: $P(G|F^{(i)})$, where G is the query feature set and $F^{(i)}$ is the feature set indexed with the ith database sound. Since $P(G|F^{(i)})$ represents the actual distribution of query features a user will produce to retrieve a database sound indexed with features $F^{(i)}$, whether or not the query is oral or file-based, this likelihood represents a model for user query behavior.

When considering possible models for query behavior, we assume that the user will at least attempt to produce a query that they consider similar to the database sound (or general class of sounds) they are trying to retrieve. The inherent similarity or difference between environmental sounds is to a significant extent determined by the corresponding similarity or difference between the *audio features* introduced in Section II: *loudness*, *spectral sparsity*, *spectral centroid*, *temporal sparsity*, *transient index*, and *harmonicity*. For certain sounds, dynamics in these features may also be salient. In other words, both query and database features should depend, at most, on the audio feature trajectories, $Y_{1:T_q}^{(1:P)}$ and $X_{1:T_i}^{(i,1:P)}$, where $Y_t^{(1:P)}$ is the query audio feature vector at frame t of T_q and $X_t^{(i,1:P)}$ is the ith database sound audio feature vector for frame t of T_i. That is, $G = g(Y_{1:T_q}^{(1:P)})$ and $F^{(i)} = f^i(X_{1:T_i}^{(i,1:P)})$ for suitably chosen functions $g(\cdot)$ and $f^i(\cdot)$. Our model, in fact, chooses $G = Y_{1:T_q}^{(1:P)}$ and $F^{(i)}$ containing the parameters of a *template* which encodes information about the "general trends" underlying the individual feature trajectories, $X_{1:T_i}^{(i,j)}$, for $j \in 1:P$. Both database and query feature trajectories, then, are modeled as noisy and distorted versions of this template, and the query likelihood model is represented as $P(Y_{1:T_q}^{(1:P)}|F^{(i)})$.

A. Template Construction

In constructing the template, we encode feature trends separately, on a feature-by-feature basis. That is, $F^{(i)} = F^{(i,1:P)}$ where $F^{(i,j)}$ encodes the trend for the jth feature trajectory. We consider only very simple trends: $X_{1:T_i}^{(i,j)}$ can be *constant*, *up*, *down*, *up* $\rightarrow$ *down*, or *down* $\rightarrow$ *up*, as it may be very difficult for the user to recall anything more detailed about how these features evolve over the duration of the sound. Furthermore, the overall level and scale of the trajectory carries some importance, and is particularly important for constant trends.[4] Both general trend and level/scale information can be encoded by subjecting each feature trajectory, $X_{1:T_i}^{(i,j)}$, to a set of zeroth, first, and second-order least-squares polynomial fits, with the optimal order chosen via Akaike criterion [37]. Each polynomial fit is then sampled evenly with one, three, or five *control points*, corresponding, respectively, to polynomial orders zero, one or two; according to the example of Fig. 10. For the kth control point we let $\phi^{(i,j)}(k) = [\chi^{(i,j)}(k)\ \dot{\chi}^{(i,j)}(k)]^T$, where $\chi^{(i,j)}(k)$ is the value of the fit and $\dot{\chi}^{(i,j)}(k)$ its derivative at this point. The template $F^{(i,j)}$, is then a HMM representation of how query features $Y_{1:T_q}^{(1:P)}$ advance through the sequence of con-

[4]For instance, if a user remembers a particular sound as "grainy" or "crackly," it is very important to the user that the sound has a high value of temporal sparsity, regardless of how this value may vary.

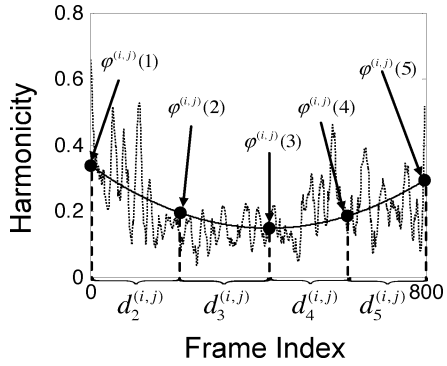

Fig. 10. Example of LS quadratic fit (solid line) and corresponding control points for a harmonicity trajectory (dotted line).

trol points (hidden states), defined by $\phi^{(i,j)}(1:K^{(i,j)})$, where $K^{(i,j)}$ is the number of control points for the jth feature and the ith sound.

In describing the HMM representation of template $F^{(i,j)}$, we must account for the possibility that query features will advance through the control point sequence $\phi^{(i,j)}(1:K^{(i,j)})$ at a variable rate. We model this *time-warping* distortion, by defining hidden states $\Lambda_t^{(i,j)} \in \phi^{(i,j)}(1:K^{(i,j)})$ that represent the control point responsible for the query observation at frame t. The advance through the control point sequence is modeled by discrete Markov process $P(\Lambda_{t+1}^{(i,j)}|\Lambda_t^{(i,j)})$, which follows the appropriate Markov transition diagram displayed in Fig. 11, depending on the order of the polynomial fit. The reason for characterizing linear and quadratic fits with more control points than the minimum required to define the curve is that we want the process of advancing through the control point sequence to represent as closely as possible the process of advancing through the optimal curve fit. The transition probabilities, $p_{ba} = P(\Lambda_{t+1}^{(i,j)} = b|\Lambda_t^{(i,j)} = a)$, are set by modeling the advance through the control point sequence as a Poisson process. For the single-jump case $(b = a+1)$, the rate parameter of the Poisson process is set to $1/d_b^{(i,j)}$, where $d_b^{(i,j)}$ is the frame difference between control points a and b in the database feature trajectory, $X_{1:T_{d,i}}^{(i,j)}$ (Fig. 10). For the two jump case $(b = a+2)$, the rate parameter is the average of $1/d_{a+1}^{(i,j)}$ and $1/d_{a+2}^{(i,j)}$. Finally, we assume that the initial hidden state corresponds to the first control point with probability one, i.e., $P(\Lambda_1^{(i,j)} = \phi^{(i,j)}(1)) = 1$.

If $\Lambda_t^{(i,j)} = k$, then the value and derivative of the observed query feature $Y_t^{(j)}$ will be close to $\phi^{(i,j)}(k)$. Hence, we model the observation probability as $P(Y_t^{(j)}|\Lambda_t^{(i,j)}) = P(\tilde{Y}_t^{(j)}|\Lambda_t^{(i,j)})$, where

$$\tilde{Y}_t^{(j)} = \left[\tilde{y}_t^{(j)} \quad \dot{\tilde{y}}_t^{(j)}\right]^T \tag{24}$$

and $\tilde{y}_t^{(j)}$ is a 41-point, fourth-order, Savitzky–Golay [38] smoothed version of $Y_t^{(j)}$ and $\dot{\tilde{y}}_t^{(j)}$ the smoothed derivative. $P(\tilde{Y}_t^{(j)}|\Lambda_t^{(i,j)})$ is given as follows:

$$P(\tilde{Y}_t^{(j)}|\Lambda_t^{(i,j)} = k) = \mathcal{N}(\phi^{(i,j)}(k), \Sigma^{(i,j)}) \tag{25}$$

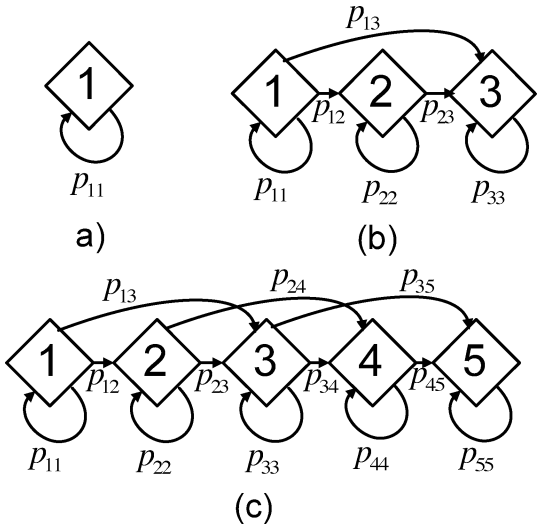

Fig. 11. Markov transition diagrams for $\Lambda_t^{(i,j)}$ under the three possible polynomial fits of the feature trajectories.

The covariance matrix, $\Sigma^{(i,j)} \in \mathbb{R}^{2\times 2}$, is designed to encode the joint uncertainties in value and derivative between time-aligned versions of the template and Savitzky–Golay smoothed observation trajectories. In our work, the covariance matrix is estimated as the sample covariance of the residual vector $e_t^{(i,j)} = \tilde{X}_t^{(i,j)} - [\chi_t^{(i,j)} \dot{\chi}_t^{(i,j)}]^T$, i.e., the difference of the smoothed feature trajectory and the chosen polynomial fit. The HMM template $F^{(i,j)}$ for sound i and feature j is then composed of the control point sequence $\phi^{(i,j)}(1:K^{(i,j)})$, covariance matrix $\Sigma^{(i,j)}$, and the associated transition probabilities.

B. Query Likelihood Model

Since the templates are fit independently to each feature, we model the query feature trajectories as conditionally independent given the templates. Furthermore, we assume that each trajectory depends only on the template information for its corresponding feature

$$P\left(Y_{1:T_q}^{(1:P)}|F^{(i)}\right) = \prod_{j=1}^{p} P\left(Y_{1:T_q}^{(j)}|F^{(i,j)}\right). \tag{26}$$

Thus, given query feature set $Y_{1:T_q}^{(1:P)}$ the likelihood from (26) can be computed in linear time using the forward algorithm [39], for each database template $F^{(i,j)}$.

VI. Fast Retrieval via Efficient Cluster-Based Indexing

Retrieval consists of obtaining the R database sounds with the greatest query likelihoods. Unfortunately, exact retrieval requires as many likelihood evaluations as there are sounds in the database, which is problematic in the case of monitoring applications that involve thousands of sounds. Instead, we have developed a cluster-based indexing strategy that effectively reduces the number of likelihood evaluations per query. Our strategy is based on partitioning the database into clusters for which, given any pair of database sounds in different clusters, if

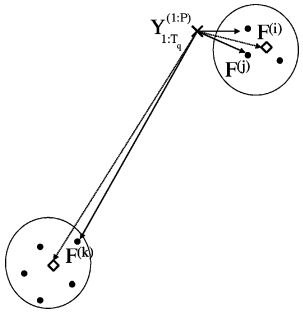

Fig. 12. Metric space using $D(F^{(i)}, F^{(k)})$ extended to incorporate query features $Y^{1:P}_{1:T_q}$. The $F^{(i)}$ are plotted with solid circles, the $Y^{(1:P)}_{1:T_q}$ with an "×," and the cluster centroids with diamonds.

the query likelihood for one sound is large, the query likelihood for the other sound will be small. In performing the query, we would restrict our search to sounds in the first cluster. Applying this clustering recursively, our retrieval strategy would have logarithmic, rather than linear, complexity in the number of database sounds.

In developing our strategy, we first present three challenges that provide necessary conditions for an effective cluster-based indexing system. First, we must construct a discriminative space based on a pseudo-metric $D(F^{(i)}, F^{(k)})$, for which the value will be small only when both $P(Y^{(1:P)}_{1:T_q}|F^{(i)})$ and $P(Y^{(1:P)}_{1:T_q}|F^{(k)})$ are large, i.e., sounds indexed by $F^{(i)}$ and $F^{(k)}$ are both likely matches to the query sound described by feature trajectory $Y^{(1:P)}_{1:T_q}$. These relations should hold as well if $F^{(k)}$ and $F^{(i)}$ are interchanged—we solve this issue by constructing the pseudo-metric to be symmetric. Second, we must develop a way of clustering the database using $D(F^{(i)}, F^{(k)})$, so that the value of this pseudo-metric is smaller between sounds in the same cluster than between sounds in different clusters. Third, we must develop an efficient way to assign the query to a particular cluster so that the query likelihood will be large only for sounds in that cluster, and likelihood evaluations for sounds in other clusters can be skipped.

If we assume $D(F^{(i)}, F^{(k)})$ is a metric, we can easily address all challenges as shown in Fig. 12. Regarding the first challenge, $D(F^{(i)}, F^{(k)})$ is chosen to be both likelihood based, and as close to a metric as possible. Concerning the second challenge, a standard clustering approach, such as k-means or spectral clustering [40], can utilize $D(F^{(i)}, F^{(k)})$ to determine the clusters. Regarding efficient assignment of query to cluster, we first represent each cluster as a template similar to those described in Section V-A. Then, only the cluster template which exhibits the largest likelihood with respect to the query observations $Y^{(1:P)}_{1:T_q}$, will have the sounds belonging to that cluster ranked in terms of likelihood and returned to the user.

We proceed in a manner analogous to that of Fig. 12, as follows. Let $X^{(i,1:P)}_{1:T_i}$ be the feature trajectory computed for the database sound indexed with $F^{(i)}$, and $L(i,k)$ a corresponding query log likelihood with respect to $F^{(k)}$

$$L(i,k) = \log\left[P\left(X^{(i,1:P)}_{1:T_i}|F^{(k)}\right)\right]. \tag{27}$$

Similar to [41], we can define a *semi-metric* $D(F^{(i)}, F^{(k)})$ based on these query likelihoods, as follows:

$$D(F^{(i)}, F^{(k)}) = L(i,i) + L(k,k) - L(i,k) - L(k,i). \tag{28}$$

It is easily verified that the following properties hold: *symmetry*: $D(F^{(i)}, F^{(k)}) = D(F^{(k)}, F^{(i)})$ and *non-negativity*: $D(F^{(i)}, F^{(k)}) \geq 0$. Regarding *distinguishability*, it is true that $D(F^{(i)}, F^{(k)}) = 0$ if $F^{(i)} = F^{(k)}$.[5] With these properties, $D(F^{(i)}, F^{(k)})$ is a *semi-metric*. It is not a metric, because it does not satisfy the triangle inequality.

Nonetheless, $D(F^{(i)}, F^{(k)})$, as constructed via (27), (28), will in most cases address the challenges outlined at the beginning of this section. Let $F^{(i)}$ be such that $P(Y^{(1:P)}_{1:T_q}|F^{(i)})$ is large; that is, the likelihood surface has a strong peak in the query feature space, and $Y^{(1:P)}_{1:T_q}$ is near that peak. Furthermore, $L(i,i)$ and $L(k,k)$ are also large, since the feature trajectory used to generate a template will also be near the peak in the likelihood surface. If $D(F^{(i)}, F^{(k)})$ is small, then, we must have $L(i,i) \approx L(i,k)$ and $L(k,k) \approx L(k,i)$, which indicates that the likelihood models become interchangeable, at least near peaks. Hence, it is very likely that $P(Y^{(1:P)}_{1:T_q}|F^{(k)})$ will also be large. On the other hand, suppose $D(F^{(i)}, F^{(k)})$ is large; either $|L(i,i) - L(i,k)|$ or $|L(k,k) - L(k,i)|$ must be large. In this case, the likelihood surfaces for templates $F^{(i)}$ and $F^{(k)}$ exhibit peaks in different regions of the feature space; thus, if $P(Y^{(1:P)}_{1:T_q}|F^{(i)})$ is large, we expect $P(Y^{(1:P)}_{1:T_q}|F^{(k)})$ to be small.

Considering the second (clustering) challenge, we compared the sound clustering performance of the k-means, non-negative matrix factorization, and spectral clustering algorithms in [42]. Based on these results, we developed a modified version of the Ng–Jordan–Weiss (NJW) spectral clustering algorithm [40], as discussed in the following section.

A. Modified Spectral Clustering

Given N database-sound feature sets, $F^{(1:N)}$, and (semi-metric) distance $D(F^{(i)}, F^{(j)})$, the NJW spectral clustering algorithm operates on an $N \times N$ *affinity matrix* with the i,j element as follows: $A(i,j) = e^{-D^2(F^{(i)}, F^{(j)})/\sigma^2}$. Here, σ is a scaling factor, and the closer $F^{(i)}$ and $F^{(j)}$ are in terms of $D(F^{(i)}, F^{(j)})$, the larger the affinity. In our adaptation of the NJW algorithm, we make the following modifications, inspired by the "self-tuning" approaches of [43]. First, we introduce *local scaling* [43] into the affinity matrix

$$A(i,j) = e^{-\frac{D^2(F^{(i)}, F^{(j)})}{\sigma_i \sigma_j}} \tag{29}$$

where $\sigma_i = D(F^{(i)}, F^{(i_M)})$ and i_M is the Mth nearest neighbor of sound i (σ_j is defined similarly). Local scaling offers more flexibility in cases where database feature sets exhibit multi-scale behavior, such as a concentrated cluster embedded in one that is more diffuse [43]. Concentrated clusters can arise when there are many equivalent sounds, for instance footsteps, while

[5]It is possible that $F^{(i)} \neq F^{(k)}$ but $L(i,i) = L(i,k)$ and $L(k,k) = L(k,i)$, which will force $D(F^{(i)}, F^{(k)}) = 0$; however, this event will occur with probability zero unless there is degeneracy in the model structure.

1.) Define $\mathbf{B}$ to be the diagonal matrix with $B(i,i) = \sum_{j=1}^{N} A(i,j)$ and construct the matrix $\mathbf{J} = \mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2}$.

2.) Find the K largest eigenvectors of $\mathbf{J}$ where K is the largest cluster number, snd stack the eigenvectors in columns to form the $N \times K$ matrix $\mathbf{V}$.

3.) Re - normalize each row of $\mathbf{V}$ to be of unit length to form the $N \times K$ matrix $\mathbf{U}$, with elements $U(i,j) = V(i,j)/\left(\sum_j V^2(i,j)\right)^{1/2}$.

4.) Loop through each $k \in [1, K]$, where k is the number of clusters. For k clusters use the first k columns of the matrix $\mathbf{U}$ to form a new matrix $\mathbf{U}'$. Treat each row of $\mathbf{U}'$ as a data point in $\mathbb{R}^k$ and cluster according to a Gaussian mixture model (GMM) using EM.

5.) Assign sound i to cluster m if the ith row of $\mathbf{U}'$ was assigned to cluster m. Calculate the BIC value under the assumption of k clusters.

6.) Choose the optimal number of clusters k_{opt} to be the k which minimizes the corresponding BIC value.

Fig. 13. Modified NJW spectral clustering algorithm.

diffuse clusters can result from sounds in more flexible categories, such as machine sounds.

Second, we use the BIC [44] to automatically choose the number of clusters at each level of the process. Even if there exist distinct semantic categories, for instance *footstep*, *machine*, and *water*, the diversity of sounds within each category (in particular, the *water* category) may induce a greater number of clusters than there are categories, and this number is difficult to predict in advance.

Given the affinity matrix constructed via (29), our adaptation of NJW spectral clustering is shown in Fig. 13.

In Fig. 13, step 5), the BIC value is calculated as follows:

$$\mathbf{BIC} = -2 \times \log(P(V|\Theta)) + I_k \times \log(N). \tag{30}$$

Here, I_k is the number of free parameters to be estimated for the GMM with k Gaussian components, $\Theta = \{\theta_1, \ldots, \theta_k\}$ denotes the collection of GMM parameters estimated via EM, and $V = \{u_1^k, \ldots, u_N^k\}$ is the collection of data points u_i^k, where u_i^k represents the ith row of matrix $\mathbf{U}'$ with dimension k. We determine $P(V|\Theta)$ according to

$$P(V|\Theta) = \prod_{i=1}^{N} \sum_{j=1}^{k} \alpha_j P_j(u_i^k|\mu_j, \Sigma_j) \tag{31}$$

where α_j is the weight for the jth Gaussian component in the GMM, and $P_j(u_i^k|\mu_j, \Sigma_j)$ is the likelihood of u_i^k evaluated with respect to the jth Gaussian component. Traditionally, the number of clusters are automatically selected via the eigenvalues or eigenvectors of the affinity matrix [43], however, we found this method tended to highly overestimate the number of clusters, when compared with the algorithm in Fig. 13. This is most likely due to the non-separability of real-world sound clusters, which we account for by using the GMM procedure in step 4), as opposed to k-means.

B. Cluster Template Construction

Recall that the third and final challenge for efficient cluster-based indexing is to develop a strategy for preassigning a query to a particular cluster, so that likelihood evaluations for sounds in other clusters can be skipped. One plausible strategy is as follows. First, we precompute for each cluster the *marginalized* query likelihood with respect to a uniform prior over the sounds in the cluster. Second, we assign the query to the cluster with the highest marginalized likelihood. While this strategy certainly guarantees that high likelihood sounds will be retrieved, it still requires query likelihood evaluations for each sound in the database.

As such, we need a query likelihood model for an entire cluster that "behaves" sufficiently like the marginalized likelihood, without incurring the expense of individual query likelihood computations. We at least attempt to capture the common perceptual characteristics of all sounds in the cluster. First, we resample all feature trajectories to a common length (the geometric mean of all lengths of sounds in the cluster) then within each feature, we treat all trajectories over all sounds in the cluster as independent and identically distributed (i.i.d.) observations from a common polynomial model and perform the polynomial fit accordingly. That is, let T be the common length and $X_{1:T}^{(i,j)}$ the jth *resampled* feature trajectory for the ith sound. We concatenate all trajectories for the kth cluster, jth feature, into the vector $C_k^{(j)} = \mathbf{vec}\{X_{1:T}^{(i,j)}\}_{i \in \mathcal{N}_k}$, where $\mathcal{N}_k$ is the index set for cluster k, and use the process as described in Section V-A to extract the appropriate (constant, linear, or parabolic) polynomial fit.

The single-feature model is otherwise identical to that developed in Section V-A, i.e., all of the concatenated feature trajectories for a cluster are represented as a single HMM with parameters determined based on the least squares polynomial fit. We denote the resultant query likelihood as $P(Y_{1:T_q}^{(j)}|F_C^{(k,j)})$, where the *cluster template* $F_C^{(k,1:P)}$ consists of the collection of HMM parameters for each feature $j \in 1:P$. Similarly, the overall cluster/query likelihood model factors as a product distribution over single-feature likelihoods.

We precompute and store the templates $F_C^{(k,1:P)}$. Given a query, we first evaluate the cluster/query likelihood (i.e., the query features are the HMM observation sequence and the likelihood is evaluated with respect to the cluster template model $F_C^{(k,1:P)}$) for each cluster. All clusters except the one maximizing the cluster/query likelihood are eliminated from future consideration. We then evaluate (26) for each sound within the maximum-likelihood cluster. For very large audio databases, we can make this procedure recursive, resulting in a number of likelihood evaluations that is logarithmic in the database size.

VII. Example-Based Query Results

We have evaluated the likelihood-based retrieval algorithm discussed in the previous sections using two audio databases, one containing automatically segmented sounds, and another

containing manually segmented sounds. The database of automatically segmented sounds consists of all sound events greater than one second in length (71 outdoor sounds and 111 indoor sounds) obtained from the approximately two hours of continuous recordings discussed in Section III. The manually segmented sound database consists of 155 indoor sounds and 145 outdoor sounds, with some of the sounds taken from the same continuous recordings as the automatically segmented data set. Although we have no ground truth labels, to provide an idea of the types of sounds we mention that each subset can be loosely partitioned into six semantic categories: *machine*, *speech*, *rhythmic*, *scratchy*, *water*, and *whistle/animal*.

A. Experimental Setup

Our study involved ten users, adults ages 23–35 with no known hearing impairments. Each user was presented with ten example audio queries and asked to rank either all 300 sounds from the manually segmented database, or all 182 sounds from the automatically segmented database as relevant or non-relevant to the query. Thus, five user rankings for each query/database sound combination were obtained. The same ten queries were used for both databases, and consisted of five sounds that were present in the manually segmented test database and five that were not, while none of the queries were present in the automatically segmented database. Additionally, the ten queries were a combination of *indoor/outdoor* sounds from all of the semantic categories mentioned above. The volunteers listened to sounds on personal computer speakers using a well known computer media player in an acoustically isolated environment. Participants were asked to fill out a worksheet recording their relevancy rankings for each of the ten example queries. The database sounds were presented in random order without revealing any category information, and with indoor and outdoor sounds mixed together, while the queries could be listened to "on-demand" throughout the ranking process. Users were allowed to complete the study at their own pace and it typically took between two and three hours (over multiple sessions) per subject to finish the study.

Given this user-determined relevance information, we evaluated retrieval performance on each database separately using standard *precision*, *recall*, and *average precision* criteria as well as the average number of likelihood computations required to rank all sounds in the database. For a given query we denote by F_{Rel} the set set of relevant sounds, and $|F_{\mathrm{Rel}}|$ as the number of relevant sounds for that query. Assuming N sounds in a database are ranked in order of decreasing likelihood for a given query, recall, and precision are computed by truncating the list to the top n sounds, and counting the number of relevant sounds, denoted by $|F_{\mathrm{Rel}}^{(n)}|$. We can then define $\mathrm{recall} = (|F_{\mathrm{Rel}}^{(n)}|)/(|F_{\mathrm{Rel}}|)$ and $\mathrm{precision} = (|F_{\mathrm{Rel}}^{(n)}|)/(n)$. Average precision is then found by incrementing n and averaging the precision at all points in the ranked list where a relevant sound is located.

As a baseline retrieval system we ranked sounds according to the Euclidean distance between the average of the smoothed feature values and their derivative for each sound event. Table III compares the mean over users and queries of the average precision values for both the proposed HMM approach and the average feature approach. We see that in all cases the HMM provided higher average precision when compared with the average feature system, suggesting that the simple dynamic trends it models are important in returning relevant sounds to users in a QBE framework. The average feature system will perform best on sounds with feature trajectories that exhibit constant trends, and the number of these sounds will vary to a certain extent based on recording conditions. In an outdoor environment, where factors such as wind and traffic noise lead to low SNR conditions, we might expect the segmentation algorithm to find longer sound events, as the background noise makes it difficult for the algorithm to decide when a sound "turns off." These longer sound events are then likely to exhibit constant feature trends, as they might be the combination of several events. Conversely, in a high SNR indoor environment more accurate segmentation performance might lead to shorter sound events containing the simple dynamic trends modeled by the HMM. This might explain why the HMM retrieval system performs best on automatically segmented indoor sounds, and manually segmented outdoor sounds (the data sets with fewer constant feature trends), while the average feature retrieval system performs best on manually selected indoor sounds and automatically selected outdoor sounds (the data sets with more constant feature trends). We now examine in detail the influence of *cluster-based indexing* on retrieval performance for the larger manually segmented database.

TABLE III
MEAN AVERAGE PRECISION FOR MANUALLY AND AUTOMATICALLY SEGMENTED SOUNDS

Segmentation Type	Indoor		Outdoor	
	HMM	Average Feature	HMM	Average Feature
Automatic	0.424	0.247	0.282	0.261
Manual	0.382	0.328	0.303	0.247

B. Clustering Performance

For the manually segmented indoor and outdoor databases we compared performance measures across two cases: *exhaustive search* and *cluster-based indexing*. For *exhaustive search*, the query likelihood was evaluated for each sound in the database, and the top n sounds in terms of likelihood were retrieved. For *cluster-based indexing*, the method presented in Section VI was used to limit the search to only those N_C sounds indexed by the most likely cluster. When $n \leq N_C$ the sounds in this cluster were retrieved in rank order, while for $n \geq N_C$ all sounds outside of the most likely cluster were retrieved in random order.

Fig. 14 shows recall and precision curves averaged over all queries and user rankings. By examining the recall curves of Fig. 14(a) and (b), we see that for the indoor database, approximately 40% of the relevant sounds were retrieved in the top 20; for the outdoor database, this number is about 35%. From the precision curves of Fig. 14(c) and (d) we see that, for the indoor database, more than 60% of the top ten returned sounds were marked as relevant, whereas for the outdoor database, this number is 40%. We conjecture that performance is better with the indoor database because the outdoor sounds generally contain more noise, or overlap between sound events. Considering the diverse perceptual quality of environmental

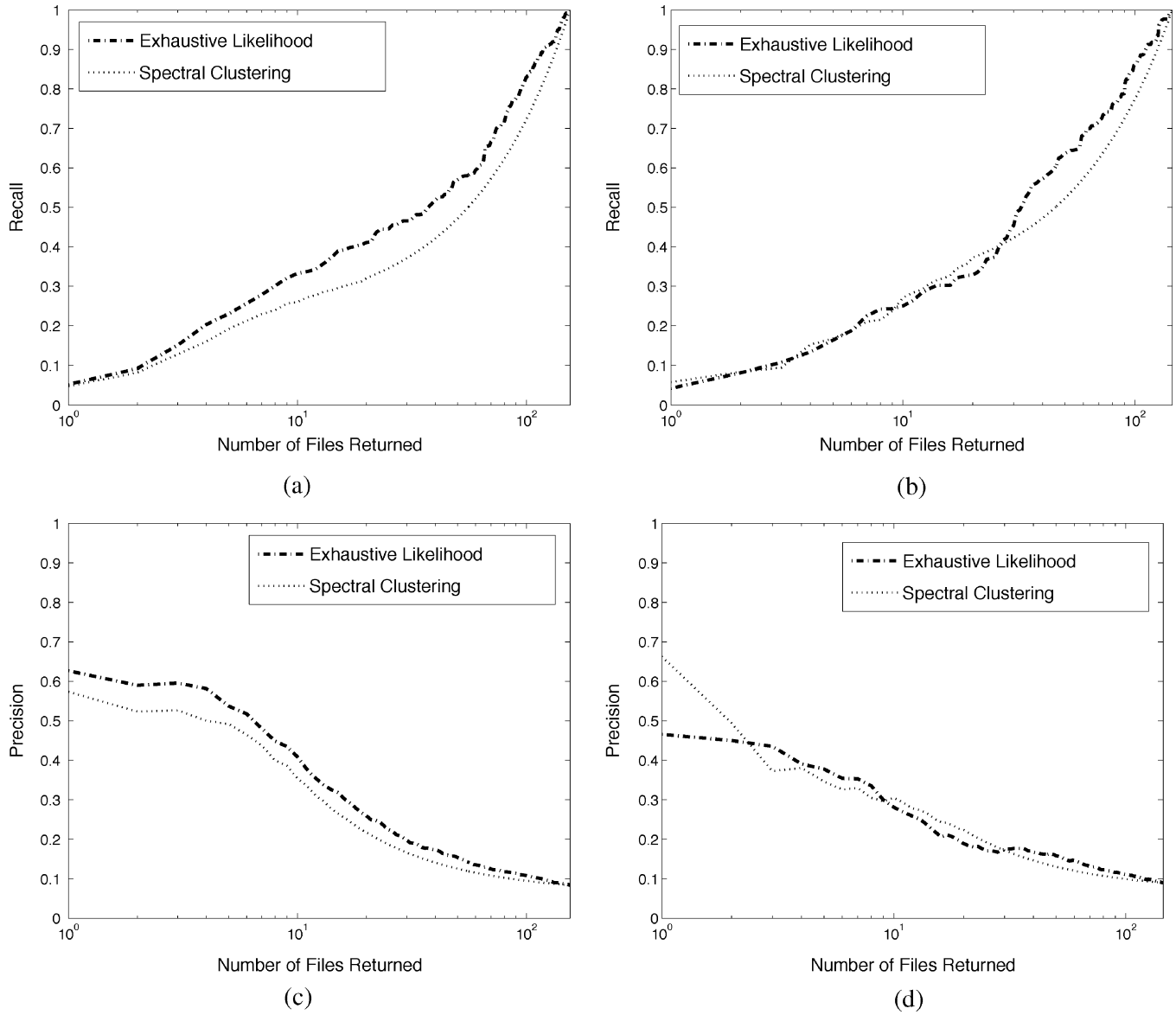

Fig. 14. Recall and precision curves averaged over ten example queries and five user relevancy rankings for indoor and outdoor sounds. (a) Recall (Indoor). (b) Recall (Outdoor). (c) Precision (Indoor). (d) Precision (Outdoor).

and natural sounds however, overall results in both cases seem quite promising. We also notice from Fig. 14(b) and (d) that the cluster-based indexing actually improves recall and precision for the top 20 retrieved outdoor sounds, rather than causing a slight detriment as expected. We conjecture that this improvement lies in the fact that clustering segregates perceptually relevant sounds from non-relevant sounds; however, further studies need to be performed before we can conclude this. As expected, clustering begins to degrade overall performance much beyond 20 retrieved sounds as sounds outside of the chosen cluster are returned randomly; however, we anticipate that in many applications the user will wish to retrieve less than 20 sounds.

As mentioned previously, average precision for an exhaustive search is simply the average of the precision values at all points in the ranked list where a relevant sound is located. To compare cluster based indexing and exhaustive search in terms of average precision we must slightly modify the average precision criterion in the cluster-based indexing case, because sounds are returned in random order outside of the first cluster. Thus the average precision becomes

$$\text{Avg}p = \frac{1}{|F_{\text{Rel}}|}\left[\sum_{n=1}^{N_C}\frac{|F_{\text{Rel}}^{(n)}|}{n}\mathbf{1}_{F_{\text{Rel}}}^{(n)} + \sum_{n=N_C+1}^{N} P(n)\times\frac{|F_{\text{Rel}}| - \left|F_{\text{Rel}}^{(N_C)}\right|}{N-N_C}\right] \quad (32)$$

where $\mathbf{1}_{F_{\text{Rel}}}^{(n)}$ is an indicator function equal to one if the sound at position n in the rank list belongs to F_{Rel} and zero otherwise, and $P(n) = (1)/(n)(|F_{\text{Rel}}^{(n)}|+(|F_{\text{Rel}}|-|F_{\text{Rel}}^{(N_C)}|)/(N-N_C))$ is the precision value at position $n \geq N_C$ in the rank list, i.e., the precision after all sounds within the chosen cluster have been

TABLE IV
MEAN AVERAGE PRECISION FOR INDOOR AND OUTDOOR QUERIES

Query Type	Indoor		Outdoor	
	Exhaustive	Spectral Clustering	Exhaustive	Spectral Clustering
Indoor Queries	0.5677	0.5038	0.1474	0.3058
Outdoor Queries	0.1258	0.1108	0.4173	0.4345

returned and remaining sounds are returned randomly. For exhaustive search the second term in the sum of (32) is unnecessary as $N_C = N$.

Table IV displays the mean average precision for both the exhaustive and cluster-based retrieval schemes using both the indoor and outdoor sound database and query sets. From Table IV, we note that for the more noisy outdoor sound database, retrieval performance is improved for both indoor and outdoor queries by using spectral clustering as certain irrelevant sounds are removed (by not being in the selected cluster) from consideration of a high position in the ranked list. We also note a general loss in performance using indoor queries on the outdoor sound database and vice-versa as the sound types in these cases might be very different. For example, the outdoor database contains several recordings of birds, which are obviously not present in the indoor database, so using a bird query might return sounds such as laughter, crying, etc., which were clearly not labeled as relevant to the bird query by the user.

To put the results of Table IV in perspective we briefly review the use of the average precision criterion in audio information retrieval tasks. Much work in information retrieval follows the NIST Text REtrieval Conference (TREC), which in 1997 began a spoken document retrieval task [45] where an audio database of speech documents was searched by text concept. In this contest the mean average precision values ranged between approximately 0.1 and 0.55 [45]. Average precision is also used in music information retrieval to compare the performance of systems used to detect cover songs [46] with values ranging from 0.0017 to 0.521. Perhaps the application most similar to ours is [47], where both MFCC features and semantic information were used to retrieve general audio data from the BBC sound effects collection, where two sounds were considered relevant if they came from the same CD of the collection. The results reported in [47] were a mean average precision of 0.165 using MFCC information alone and 0.186 when semantic information was included. Therefore, we can conclude that our mean average precision values ranging from 0.1108 to 0.5038 when spectral clustering is used, and between 0.1258 to 0.5677 for exhaustive search are quite good, and in the best case, can roughly be interpreted that for every relevant sound retrieved our system retrieves only a single irrelevant sound along with it.

C. BIC Determination of Number of Clusters

Fig. 15 plots BIC versus number of clusters, where the minimum BIC value corresponds to the optimum number of clusters. A single point on the BIC curve represents the average of ten BIC values each corresponding to a different random starting point of the EM algorithm. From Fig. 15 we see that for the indoor database, the optimum number of clusters was chosen as 16 and for the outdoor database 14. As we stated earlier, both databases have six semantic categories; thus, the BIC can be considered as overestimating the "correct" number of clusters. However, at least part of this overestimate can be explained by the fact there exist sounds within each semantic category that are perceptually distinct, for instance a printer sound, a vacuum cleaner sound, and an elevator sound are all machine sounds with very different perceptual characteristics.

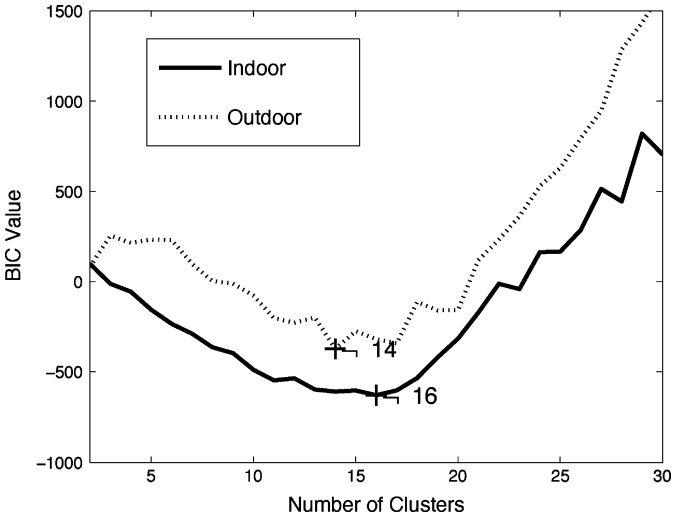

Fig. 15. BIC value versus number of clusters.

D. Computational Cost

Finally, we compare average computational costs for the cluster-based indexing versus exhaustive search in terms of the number of query likelihood evaluations. For exhaustive search, the number of likelihood evaluations equals the number of sounds in the sound set; for cluster-based indexing, the number of evaluations is $N_C + K$, where N_C is the number of sounds in the chosen cluster, K is the number of clusters, and N_C is obtained by averaging over the ten example queries in addition to using all 300 database sounds as queries for computational cost analysis only. For indoor sounds, about six times as many evaluations are required for exhaustive search versus cluster-based indexing (155 versus 25.80); for outdoor sounds, over four times as many evaluations are necessary (145 versus 35.10). Hence, in applications where less than 20 sounds are to be retrieved, we have observed advantages in *both* retrieval performance and computational cost, the cost advantage being substantial. We have remarked that for much larger databases, the clustering can be made recursive and thus lead to even greater computational savings. So far, such a large-scale application has not been tested, due to the difficulty in obtaining the tens of thousands of subject-hours required to establish relevancies for all of the sounds.

VIII. CONCLUSION AND FUTURE WORK

In order to characterize the sound activity in fixed spaces we have presented a complete system for segmentation, indexing, and retrieval of natural and environmental sounds. By choosing a compact, yet general acoustic feature set as the basis for our segmentation and retrieval algorithms, we developed a general approach applicable to a broad range of sound classes. Using dynamic probabilistic models, the feature trajectories

from continuously recorded audio are examined to segment the audio into distinct events or auditory streams, as the examples in Sections III and IV illustrate. Furthermore, our approach can be extended to work with sparse microphone arrays distributed throughout spaces where a single microphone is insufficient. Once segmented, every sound event is stored in a database and indexed with a probabilistic model, which is used for likelihood-based retrieval.

Since every sound event is indexed with its own probabilistic model, which requires inference evaluation every time a query is presented, efficient retrieval in a query-by-example paradigm can be difficult. We propose to overcome this problem using a modified spectral clustering algorithm where the similarity between two indexed sound events is related to the distance between their corresponding probabilistic models. The number of clusters in a given collection of segmented sound events is automatically determined by appending a BIC for Gaussian mixture models into the spectral clustering algorithm. Furthermore, these clusters can be used as a generalized environmental sound classification system, where neither the class types nor the number of classes are known *a priori*. The clustering system was shown to provide large savings in search complexity, while minimally impacting retrieval accuracy on two test databases captured under different recording conditions.

One possible enhancement to our system would be to extend our query-by-example system to the query-by-humming domain, i.e., users can search the database with their voice. Our current retrieval system will work for human voice generated queries under certain conditions, but in dealing with natural and environmental sounds, there are certain acoustic qualities the human voice cannot imitate accurately, if at all. Thus, beginning with the feature set of Section II, we would hope to establish through extensive user studies, those features of environmental sounds that the human voice can accurately imitate, and those which cannot be mimicked. We would also like to investigate whether cross mappings occur, e.g., users are very likely to use pitch to mimic a sound quality that is actually due to spectral centroid. We then hope to use this information to improve the user experience for oral retrieval of natural and environmental sounds. Further study of our distortion-aware models [48] for accommodating scale/level distortion in acoustic feature trajectories might also help improve retrieval performance for oral queries.

Another clear improvement to our system would be the incorporation of semantic information. Although the methods described in Section V can be used to cluster sounds based on perceptual similarity, sounds may contain explicit semantic information or connotations that connect them, despite the differences of their sonic qualities. A sound of a ship, for example, may not contain qualities that cluster it with sounds of water, despite the intuitive closeness of these two sounds.

Semantic information can be represented through external ontologies such as WordNet and ConceptNet [49] where a metric such as shortest path distance can be used to measure distances between concepts. By combining these semantic distances with measures of acoustic feature similarity we can construct an ontological framework [50], where users provide information linking sounds to other multimedia and concepts. Including semantic information in this fashion can assist in generalizing the audio information retrieval process beyond query-by-example, and tailor performance for specific user communities. In cases where the retrieval process is not only oral, for example, semantic relations between sounds and segments of other activity, such as physical gesture or text-based query, can be used to create a more robust action-based retrieval system.

ACKNOWLEDGMENT

The authors would like to thank T. Rikakis, A. Fink, J. Liu, and our anonymous reviewers for their helpful comments and suggestions.

REFERENCES

[1] R. Schafer, *The Soundscape*. Rochester, VT: Destiny Books, 1968.
[2] B. Truax, *Acoustic Communication*. Norwood, NJ: Ablex Publishing, 1984.
[3] A. Bregman, *Auditory Scene Analysis: The Perceptual Organization of Sound*. Cambridge, MA: MIT PRess, 1990.
[4] A. Harma, M. F. McKinney, and J. Skowronek, "Automatic surveillance of the acoustic activity in our living environment," in *IEEE Int. Conf. Multimedia and Expo*, Amsterdam, The Netherlands, Jul. 2005.
[5] P. Hedfors and P. Grahn, R. Schafer and H. Jarviluoma, Eds., "Soundscapes in urban and rural planning and design—A brief communication of a research project," in *Northern Soundscapes: Yearbook of Soundscape Studies*, vol. 1, pp. 67–82.
[6] D. P. W. Ellis and K. Lee, "Minimal-impact audio-based personal archives," in *Proc. 1st ACM Workshop Continuous Archiving and Recording of Personal Experiences CARPE-04*, New York, Oct. 2004.
[7] J. Gemmell, G. Bell, and R. Lueder, "MyLifeBits: A personal database for everything," *Commun. ACM*, vol. 49, no. 1, pp. 88–95, 2006.
[8] D. F. Rosenthal and H. G. Okuno, *Computational Auditory Scene Analysis*. Mahwah, NJ: Lawrence Erlbaum Associates, 1998.
[9] R. Andre-Obrecht, "A new statistical approach for automatic segmentation of continuous speech signals," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 36, no. 1, pp. 29–40, Jan. 1988.
[10] A. T. Cemgil, "Bayesian Music Transcription," Ph.D. dissertation, Radboud Univ. of Nijmegen, Nijmegen, The Netherlands, 2004.
[11] H. Thornburg, "Detection and modeling of transient audio signals with prior information," Ph.D. dissertation, Stanford Univ., Stanford, CA, 2005.
[12] L. Lu, R. Cai, and A. Hanjalic, "Audio elements based auditory scene segmentation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Toulouse, France, 2006.
[13] G. Wichern, H. Thornburg, B. Mechtley, A. Fink, K. Tu, and A. Spanias, "Robust multi-feature segmentation and indexing for natural sound environments," in *Proc. IEEE/EURASIP Int. Workshop Content-Based Multimedia Indexing (CBMI)*, Bordeaux, France, 2007, pp. 69–76.
[14] A. Dielmann and S. Renals, "Automatic meeting segmentation using dynamic Bayesian networks," *IEEE Trans. Multimedia*, vol. 9, no. 1, pp. 25–36, Jan. 2007.
[15] D. Zhang, D. Gatica-Perez, S. Bengio, and I. McCowan, "Modeling individual and group actions in meetings with layered HMMs," *IEEE Trans. Multimedia*, vol. 8, no. 3, pp. 509–520, May 2006.
[16] C. J. V. Rijsbergen, *Information Retrieval*. London, U.K.: Butterwoths, 1979.
[17] A. S. Durey and M. A. Clements, "Melody spotting using hidden Markov models," in *Proc. Int. Symp. Music Inf. Retrieval (ISMIR),*, Bloomington, IN, 2001.
[18] J. Shifrin, B. Pardo, C. Meek, and W. Birmingham, "HMM-based musical query retrieval," in *Proc. 2nd ACM/IEEE-CS Joint Conf. Digital Libraries*, Portland, OR, 2002.
[19] S. Shalev-Shwartz, S. Dubnov, N. Friedman, and Y. Singer, "Robust temporal and spectral modeling for query by melody," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, Tampere, Finland, 2002.
[20] E. Wold, T. Blum, D. Keislar, and J. Wheaton, "Content-based classification, search, and retrieval of audio," *IEEE Multimedia*, vol. 3, no. 3, pp. 27–36, 1996.

[21] M. F. McKinney and J. Breebaart, "Features for audio and music classification," in *Proc. 4th Int. Conf. Music Inf. Retrieval*, Baltimore, MD, Oct. 2003.
[22] 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.
[23] H. G. Kim, N. Moreau, and T. Sikora, *MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval*. West Sussex, U.K.: Wiley, 2005.
[24] J. O. Smith III and J. S. Abel, "Bark and ERB bilinear transforms," *IEEE Trans. Speech Audio Process.*, vol. 7, no. 6, pp. 697–708, Nov. 1999.
[25] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization," in *Proc. National Academy Sci.*, 2003, vol. 100, no. 5, pp. 2197–2202.
[26] *Speech Processing, Transmission and Quality Aspects (STQ); Distributed Speech Recognition; Front-End Feature Extraction Algorithm; Compression Algorithms*, ETSI ES 201 108 v1.1.3 (2003–09), 2003, E.T.S.I. standard document.
[27] A. de Cheveigne and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *J. Acoust. Soc. Amer.*, vol. 111, no. 4, pp. 1917–1930, 2002.
[28] J. L. Goldstein, "An optimum processor theory for the central formation of the pitch of complex tones," *J. Acoust. Soc. Amer.*, vol. 54, no. 6, pp. 1496–1516, 1973.
[29] H. Thornburg and R. J. Leistikow, "A new probabilistic spectral pitch estimator: Exact and MCMC-approximate strategies," in *Lecture Notes in Computer Science 3310*, U. K. Wiil, Ed. New York: Springer-Verlag, 2005.
[30] H. Thornburg, R. Leistikow, and J. Berger, "Melody extraction and musical onset detection via probabilistic models of STFT peak data," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 4, pp. 1257–1272, May 2007.
[31] F. Gustaffsson, *Adaptive Filtering and Change Detection*. New York: Wiley, 2001.
[32] V. Pavlovic, J. M. Rehg, and T. Cham, "A dynamic Bayesian network approach to tracking learned switching dynamic models," in *Proc. Int. Workshop Hybrid Syst.*, Pittsburgh, PA, 2000.
[33] S. Chen and P. Gopalakrishnan, "Speaker environment and channel change detection and clustering via the Bayesian information criterion," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, 1998.
[34] D. Ellis and K. Lee, "Accessing minimal-impact personal audio archives," *IEEE Multimedia*, vol. 13, no. 4, pp. 30–38, Jul. 2006.
[35] G. Wichern, H. Thornburg, and A. Spanias, "Multi-channel audio segmentation for continuous observation and archival of large spaces," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Taipei, Taiwan, 2009, pp. 237–240.
[36] A. Doucet, N. de Freitas, K. Murphy, and S. Russell, "Rao-Blackwellised particle filtering for dynamic Bayesian networks," in *Proc. Conf. Uncertainty in Artif. Intell.*, Stanford, CA, 2000.
[37] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Autom. Control*, vol. AC-19, no. 6, pp. 716–723, Mar. 1974.
[38] A. Savitzky and M. J. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Anal. Chem.*, vol. 36, no. 8, pp. 1627–1639, 1964.
[39] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
[40] A. Y. Ng, M. Jordan, and Y. Weiss, "On spectral clustering analysis and an algorithm," in *Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, 2002.
[41] B. H. Huang and L. R. Rabiner, "A probabilistic distance measure for hidden Markov models," *AT&T Tech. J.*, vol. 64, no. 2, pp. 1251–1270, 1985.
[42] J. Xue, G. Wichern, H. Thornburg, and A. S. Spanias, "Fast query-byexample of environmental sounds via robust and efficient cluster-based indexing," in *IEEE Int. Conf. Acoust., Speech, Signal Process.*, Las Vegas, NV, 2008, pp. 5–8.
[43] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Adv. Neural Inf. Process. Syst.*, Whistler, BC, Canada, 2004.
[44] G. Schwarz, "Estimating the dimension of a model," *Ann. Statist.*, vol. 6, no. 2, pp. 461–464, 1978.
[45] J. S. Garofolo, C. G. P. Auzanne, and E. M. Voorhees, "The TREC spoken document retrieval track: A success story," in *Proc 8th Text REtrieval Conf. (TREC)*, Gaithersburg, MD, 1999.
[46] J. H. Jensen, M. G. Christensen, D. Ellis, and S. H. Jensen, "A tempo-insensitive distance measure for cover song identification based on chroma features," in *IEEE Int. Conf. Acoust., Speech, Signal Process.*, Las Vegas, NV, 2008.
[47] L. Barrington, A. Chan, D. Turnbull, and G. R. G. Lanckriet, "Audio information retrieval using semantic similarity," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Honolulu, HI, 2007, pp. 725–728.
[48] G. Wichern, J. Xue, H. Thornburg, and A. Spanias, "Distortion-aware query by example for environmental sounds," in *Proc. IEEE Workshop the Applicat. Signal Process. Audio Acoust. (WASPAA)*, New Paltz, NY, 2007, pp. 335–338.
[49] H. Liu and P. Singh, "Conceptnet: A practical commonsense reasoning toolkit," *BT Technol. J.*, vol. 22, no. 4, pp. 211–226, 2004.
[50] G. Wichern, H. Thornburg, and A. Spanias, "Unifying semantic and content-based approaches for retrieval of environmental sounds," in *Proc. IEEE Workshop Applicat. Signal Process. to Audio Acoust. (WASPAA)*, New Paltz, NY, 2009, pp. 13–16.

Gordon Wichern (S'08) received the B.S. and M.S. degrees in electrical engineering from Colorado State University, Fort Collins, in 2004 and 2006, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with a concentration in arts, media and engineering at Arizona State University (ASU), Tempe.

He is supported by a National Science Foundation (NSF) Integrative Graduate Education and Research Traineeship (IGERT) in arts, media and engineering. His primary research interests include signal processing, machine learning, and information retrieval. He has held internship appointments in computational finance at SAP Labs and music information retrieval at the Yamaha Center for Advanced Sound Technologies.

Jiachen Xue received the B.S. degree in software engineering from Beihang University, Beijing, China, in 2006, and the M.S. degree in electrical engineering from Arizona State University, Tempe, in 2008. He is currently pursuing the Ph.D. degree in computer and electrical engineering at Purdue University, West Lafayette, IN.

His primary research interests include environmental audio signal processing, content-based audio clustering, and information retrieval.

Harvey Thornburg (M'09) received the Ph.D. degree in electrical engineering from the Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, Stanford, CA, in 2005.

He is an Assistant Professor with a joint appointment in electrical engineering and Arts, Media, and Engineering (AME), Arizona State University (ASU), Tempe. His current research interests concern the design of experiential media systems: immersive multisensory environments situated in the physical world that respond to natural human activity at the level of meaning. He has led research efforts in multimodal human activity analysis, full-body gesture analysis, and computational auditory scene analysis. He has recently served as Co-Director of the Situated Multimedia Arts Learning Laboratory (SMALLab), a mixed-reality learning environment that has positively impacted thousands of students at local and national levels and now is Director of the Mediating Complex Systems initiative at AME/ASU.

Brandon Mechtley (S'09) received the B.S. degree in computer science from Arizona State University (ASU), Tempe, in 2007. He is currently pursuing the Ph.D. degree in computer science with a concentration in arts, media, and engineering at ASU.

He is supported by a National Science Foundation (NSF) Integrative Graduate Education and Research Traineeship (IGERT) in arts, media, and engineering and is a Science Foundation Arizona Graduate Fellowship awardee. His interests include social media, machine learning, and audio analysis and synthesis.

Andreas Spanias (S'84–M'85–SM'94–F'03) received the Ph.D. degree in electrical engineering from West Virginia University, Morgantown, in 1988.

He is a Professor in the Department of Electrical Engineering, Arizona State University (ASU), Tempe. He is also the Director of the SenSIP industry consortium. His research interests are in the areas of adaptive signal processing, speech processing, and audio sensing. He and his student team developed the computer simulation software Java-DSP (J-DSP-ISBN 0-9724984-0-0). He is author of two textbooks: *Audio Processing and Coding* (Wiley, 2007) and *DSP; An Interactive Approach* (www.lulu.com publishers).

Dr. Spanias was corecipient of the 2002 IEEE Donald G. Fink Paper Prize Award. He served as an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING and as General Co-Chair of ICASSP'99. He also served as the IEEE Signal Processing Vice-President for Conferences. He served as Distinguished Lecturer for the IEEE Signal Processing Society in 2004.