DCAR: A Discriminative and Compact Audio Representation for Audio Processing

Liping Jing, Bo Liu, Jaeyoung Choi, Adam Janin, Julia Bernd, Michael W. Mahoney, and Gerald Friedland

***Abstract*—This paper presents a novel two-phase method for audio representation, discriminative and compact audio representation (DCAR), and evaluates its performance at detecting events and scenes in consumer-produced videos. In the first phase of DCAR, each audio track is modeled using a Gaussian mixture model (GMM) that includes several components to capture the variability within that track. The second phase takes into account both global structure and local structure. In this phase, the components are rendered more discriminative and compact by formulating an optimization problem on a Grassmannian manifold. The learned components can effectively represent the structure of audio. Our experiments used the YLI-MED and DCASE Acoustic Scenes datasets. The results show that variants on the proposed DCAR representation consistently outperform four popular audio representations (mv-vector, i-vector, GMM, and HEM-GMM). The advantage is significant for both easier and harder discrimination tasks; we discuss how these performance differences across tasks follow from how each type of model leverages (or does not leverage) the intrinsic structure of the data.**

***Index Terms*—Audio representation, audio scene classification, compact representation, discriminativity, event detection.**

Manuscript received November 21, 2016; revised March 14, 2017 and April 23, 2017; accepted April 30, 2017. Date of publication May 12, 2017; date of current version November 15, 2017. This work was supported in part by the NSFC under Grant 61370129, Grant 61375062, and Grant 61632004, in part by the PCSIRT under Grant IRT201206, in part by a collaborative Laboratory Directed Research & Development grant led by the Lawrence Livermore National Laboratory (U.S. Dept. of Energy Contract DE-AC52-07NA27344), and in part by the Science and Technology Bureau of Baoding City under Grant 16ZG026. This paper was presented in part at the ACM Multimedia Conference, Amsterdam, The Netherlands, October 2016. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yi-Hsuan Yang. *(Corresponding author: Liping Jing.)*

L. Jing is with the Beijing Key Laboratory of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing 100044, China (e-mail: lpjing@bjtu.edu.cn).

B. Liu is with the College of Information Science and Technology, Agricultural University of Hebei, Hebei 071000, China, and also with the Beijing Key Laboratory of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing 100044, China (e-mail: 12112082@bjtu.edu.cn).

J. Choi is with the International Computer Science Institute, Berkeley, CA 94704 USA, and also with the Delft University of Technology, Delft 2628, The Netherlands (e-mail: jaeyoung.icsi@gmail.com).

A. Janin and J. Bernd are with the International Computer Science Institute, Berkeley, CA 94704 USA (e-mail: janin@icsi.berkeley.edu; jbernd@icsi.berkeley.edu).

M. W. Mahoney is with the International Computer Science Institute, Berkeley, CA 94704 USA, and also with the Department of Statistics, University of California, Berkeley, Berkeley, CA 94720 USA (e-mail: mmahoney@stat.berkeley.edu).

G. Friedland is with Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA 94720 USA, and also with the Lawrence Livermore National Laboratory, Livermore, CA 94550 USA (e-mail: fractor@eecs.berkeley.edu).

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

Digital Object Identifier 10.1109/TMM.2017.2703939

I. Introduction

To MEET the challenges presented by content-based retrieval of user-generated videos, we need approaches that leverage cues in all available modalities. One less explored approach is to analyze the soundtracks. But while analysis of visual content is widely studied, for example in event detection, analysis of video soundtracks has largely been restricted to specific, inherently audio-focused tasks such as speech processing and music retrieval.

However, when visual information cannot reliably identify content (e.g., due to poor lighting conditions), audio may still furnish vivid information. In other words, audio content is frequently complementary to visual content—and in addition, it offers more tractable processing. In recent years, audio processing has therefore gained greater importance in multimedia analysis [2]–[6].

In particular, multimodal approaches that use both visual and audio cues have recently gained traction [7]. However, there has not been much in-depth exploration of how best to leverage the audio information. On the audio side, due to a historical focus on carefully-curated speech and music processing corpora [3], fewer audio researchers consider the problems posed by unfiltered generic audio with varied background noises—but these problems must be addressed to build audio classifiers that can handle user-generated video. In addition, past work on video analysis has often fed audio "blindly" into a machine learner, without much consideration of how audio information is structured.

In the last few years, the focus has been changing in both event detection and audio analysis. For example, sound event detection and acoustic scene classification have been included several times in the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge at IEEE AASP [5].

Major areas of audio-based event and scene detection research include audio data representation and learning methodologies. In this work, we focus on the former, which aims to extract specific features that can refine an enormous amount of raw audio data into higher-level information about the audio signals. Section II gives an overview of current representation approaches and discusses their limitations in detail. As a brief summary, current approaches do not effectively capture signal variance within audio tracks nor local structure (for example, between Gaussian components), they risk losing information about geometric manifold structure and hidden structure within the data, they often require a lot of storage space, and they rarely leverage available information from labels.

1520-9210 © 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

In this paper, we address these issues by introducing a Discriminative and Compact Audio Representation (DCAR) to model audio information. This method is implemented in two phases. First, each audio track is modeled using a Gaussian mixture model (GMM) with several mixture components to describe its statistical distribution. This is beneficial for capturing the variability within each track and for reducing the storage required, relative to the full original set of frames. We also tested a variant in which a few components are generated for each class; we call this efficient representation based on class-aware components DCAR(c).

Second, by integrating the labels for the audio tracks and the local structure among the Gaussian components, we identify an embedding to reduce the dimensionality of the mixture components and render them more discriminative. In this phase, the dimensionality reduction task is formulated as an optimization problem on a Grassmannian manifold and solved via the conjugate gradient method. Then a new audio track can be represented with the aid of the learned embedding, which further compacts the audio information. For classification, we adopt the kernel ridge regression (KRR) method, which is compatible with the manifold structure of the data.

As we argue in Section III, DCAR/DCAR(c) represents a considerable advancement over existing popular methods in audio-based event classification. In a nutshell, the novelty of DCAR lies in its being a *compact representation* of an audio signal that *captures variability* and has *better discriminative ability* than other representations.

Our claim is supported by a series of experiments, described in Section VI, conducted on the YLI-MED and DCASE datasets. We first built binary classifiers for each pair of events in YLI-MED, and found that the proposed DCAR performed better than i-vectors on pairwise discrimination. We then delved deeper, comparing multi-event classification results for DCAR and DCAR(c) with four existing methods (including simple GMMs, HEM-GMMs, and mean/variance vectors as well as i-vectors) for events that are difficult to distinguish vs. events that are easy to distinguish. We showed that DCAR/DCAR(c) can handle both easy and hard cases; Section VI-B discusses how these results may follow from how each model leverages (or doesn't leverage) the intrinsic structure of the data. Finally, we conducted multi-event classification experiments on all ten events, again showing that DCAR and DCAR(c) are the most discriminative representations.

The event-detection results were mirrored in experiments on audio scene classification using the DCASE dataset. We again compared DCAR and DCAR(c) with i-vector, mv-vector, simple GMMs, and HEM-GMM, along with a baseline method provided for DCASE. DCAR achieved the best average results among the seven systems. Per-class, DCAR and DCAR(c) outperformed the other methods on in-vehicle scenes, while achieving competitive results on indoor and outdoor scenes.

The remainder of this paper is organized as follows: Section II surveys related audio work. Section III presents the proposed DCAR model in detail, and Section IV describes the classification with KRR. Section V describes how we applied DCAR in event detection on YLI-MED, and Section VI discusses our binary detection and multi-event classification experiments. Sections VII and VII-C discuss methods and results from audio scene–classification experiments on DCASE data. Conclusions and future work are discussed in Section VIII.

II. Related Work

Audio representations include low-level features (e.g., energy, cepstral, and harmonic features) and intermediate-level features obtained via further processing steps such as filtering, linear combination, unsupervised learning, and matrix factorization (see overview in Barchiesi *et al.,* 2015 [8]).

A typical audio representation method for event detection is to model each audio track as a vector so traditional classification methods can be easily applied. The most popular low-level features used are Mel-frequency cepstral coefficients (MFCCs) [9] and variants thereof (e.g., GTCCs [4]). However, MFCC is a short-term frame-level representation, so it does not capture the whole structure hidden in each audio signal. To address this, some researchers have used end-to-end classification methods (e.g., neural networks), for example to simultaneously learn intermediate-level audio concepts and train an event classifier [10]. Several approaches have used first-order statistics derived from the frames' MFCC features, improving performance on audio-based event detection. For example, Jin *et al.* adopted a codebook model to define audio concepts [11]. This method quantizes low-level features into discrete codewords, generated via clustering, and provides a histogram of codeword counts for each audio track (i.e., it uses the mean of the data in each cluster).

However, such methods do not capture the complexity of real-life audio recordings. For event detection, researchers have therefore modeled audio using the second-order statistical covariance matrix of the low-level MFCC features [12]–[15]. There are two ways to compute the second-order statistics. The first assumes that each audio track can be characterized by the mean and variance of the MFCCs representing each audio frame, then modeled via a vector by concatenating the mean and variance [15]; this representation can be referred to as a mean/variance vector or *mv-vector*. The other method is to model all training audio via a Gaussian mixture model and then compute the Baum-Welch statistics of each audio track according to the mixture components, as in GMM-supervector representations [12]. Again, each audio track is represented by stacking the means and covariance matrices.

However, it can be time-consuming to deal with GMMs for each audio track in later analysis steps. Turnbull *et al.* [16] proposed the HEM-GMM method, which first generates GMM components from each track, then groups the components belonging to each class and uses the cluster centers to represent that class. This reduces the number of components going into subsequent analysis. Our class-aware representation, DCAR(c), follows this approach.

An exciting area of recent work is the i-vector approach, which uses latent factor analysis to compensate for foreground and background variability [17]. The i-vector approach can be seen as an extension of the GMM-supervector. It assumes that these high-dimensional supervectors can be confined to a low-dimensional subspace; this can be implemented by applying probabilistic principal component analysis (PPCA) to the supervectors. The advantage of an i-vector is that the system learns the total variance from the training data and then uses it on new data, so that the representation of the new data has similar discriminativity to the representation of the training data. I-vectors have shown promising performance in audio-based event detection [13], [14].

In fact, many of these representations have shown promising performance, but they have some limitations with regard to audio-based event detection. For example, the signal variance within a given audio track may be large; training a Gaussian mixture model on all of the audio tracks (as in the GMM-supervector or i-vector approaches) does not capture that variability, and thus may not characterize a given event well.

The second limitation is that each mixture component consists of both a mean vector and a covariance matrix, which can introduce many more variables, and so result in high computational complexity and require a lot of storage space. The third limitation is that the covariance matrices of the mixture components in these methods are usually flattened into one supervector, which may distort the geometric manifold structure[1] within the data [18] and lose information about hidden structure. Fourth, most audio representations are derived in an unsupervised manner, i.e., they do not make use of any existing label information. But in fact, label information has been very useful for representing data in classification tasks such as image classification [19] and text classification [20]. Last but not least, these methods do not explicitly consider the local structure between Gaussian components, which may be useful for distinguishing events.

At a different level of semantic structure, i.e., individual audio concepts or noises, recent work has experimented with using neural networks to deal with large-scale datasets (e.g., [21]). However, for higher-level events and scenes, the audio data may be too limited to train a neural network sufficiently.

These drawbacks motivate us to propose a new audio representation method to capture the variability within each audio track and to characterize the distinct structures of events with the aid of valuable existing labels and local structure within the data; these characteristics of our method have significant benefits for event detection and audio-scene classification.

III. Discriminative and Compact Audio Representation

In this section, we describe our proposed two-phase audio representation method. The first phase, described in Section III-A, aims to capture the variability within each audio track. The second phase, described in Section III-B, identifies a discriminative embedding.

A. Phase 1: Characterizing Per-Track Variability

We first extract the low-level features, in this case MFCCs, from the audio tracks. Let $\mathbf{X} = \{\mathbf{X}^k\}_{k=1}^n$ denote a set of n audio tracks. Each track $\mathbf{X}^k$ is segmented into m_k frames. Each frame x_f^k ($1 \leq k \leq n$ and $1 \leq f \leq m_k$) is modeled via a vector with d-dimensional MFCC features ($d = 60$), i.e., $x_f^k \in \mathbb{R}^{60}$. (Details on how we extract the MFCCs are given in Section V-A.)

Previous work has demonstrated that second-order statistics are much more appropriate for describing complicated multimedia data [17], [22]. Therefore, we train a GMM with P components using the Expectation-Maximization algorithm for each audio track (here we call these components track-aware components)

$$\mathbf{X}^k = \{x_f^k\}_{f=1}^{m_k} \in \mathbb{R}^{d \times m_k}. \tag{1}$$

The estimated GMM components are denoted as

$$G = \{g_i\}_{i=1}^N \tag{2}$$

where $g_i = \{w_i, \mu_i, \Sigma_i\}$. When each audio track is modeled via P components, $N = nP$. Each component has its corresponding weight w_i, mean μ_i, and covariance matrix Σ_i.

Generally, covariance matrices are positive semi-definite, and can be made strictly positive definite by adding a small constant to the diagonal elements of the matrix. For convenience, we use the notation Σ to indicate a symmetric positive definite (SPD) matrix. After GMM modeling, each audio track—typically containing hundreds to thousands of frames—is reduced to a smaller number of mixture components with prior probabilities. The covariance matrices provide a compact and informative feature descriptor, which lies on a specific manifold, and obviously captures the (second-order) variability of the audio.

As we noted above, we also tested a variant, DCAR(c), in which this step generates class-aware GMM components instead of track-aware. Specifically, we first employ the regular EM to generate GMM components over all the frames of each class, and then use them to learn the embedding.

B. Phase 2: Identifying a Discriminative Embedding

In the second phase of the DCAR method, a discriminative embedding is identified by integrating the global and local structure of the training data, so that both training data and unseen new data can be re-represented in a discriminative and compact manner.

1) Overview of Phase 2: The model presented thus far ignores the global structure of the data (e.g., the valuable label information) and the local structure among the components (e.g., nearest neighbors). Meanwhile, the original feature representation is usually large (since there are 60 MFCC features, each mean vector has 60 elements, and each covariance matrix contains 60×60 elements), which may be time-consuming in later data processing. Therefore we propose a new method for generating a discriminative and compact representation from the high-dimensional mixture components. The DCAR method is summarized in Fig. 1.

Our main goal is to learn an embedding $\mathbf{W} \in \mathbb{R}^{d \times r}$ ($r < d$, where d is the number of MFCC features and r is the embedding space size) based on the GMM components of the labeled audio track ($G = \{g_i, \ell_i\}_{i=1}^N$, where ℓ_i is the label for component g_i, based on the label of the corresponding audio track from which g_i was generated). The resulting low-dimensional GMM components should preserve the important structure of the original GMM components as much as

[1]By *geometric structure*, we mean intrinsic structure within data such as affine structure, projective structure, etc.

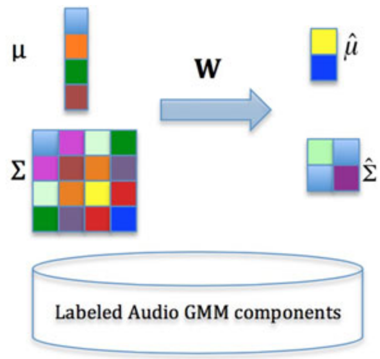


Fig. 1. Framework for generating the discriminative and compact audio representation (DCAR). The left side shows the original d-dimensional GMM components ($\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d\times d}$); the right side shows the DCAR representation with r-dimensional ($r < d$) mixture components ($\hat{\mu} \in \mathbb{R}^r$ and $\hat{\Sigma} \in \mathbb{R}^{r\times r}$).

possible. To accomplish this, we introduce an embedding $\mathbf{W}$ and define the new GMM components with mean

$$\hat{\mu} = \mathbf{W}^T \mu \tag{3}$$

and covariance matrix

$$\hat{\Sigma} = \mathbf{W}^T \Sigma \mathbf{W}. \tag{4}$$

As mentioned above, the covariance matrix Σ is SPD, i.e., $0 \prec \Sigma \in \mathcal{S}ym_d^+$. To maintain this property, i.e., $0 \prec \hat{\Sigma} \in \mathcal{S}ym_r^+$, the embedding $\mathbf{W}$ is constrained to be full rank. A simple way of enforcing this requirement is to impose orthonormality constraints on $\mathbf{W}$ (i.e., $\mathbf{W}^T\mathbf{W} = \mathbf{I}_r$), so that the embedding can be identified by solving an optimization problem on the Grassmannian manifold.

The event or scene label for each training track, which we also assign to the GMM components, can be interpreted as global structure for those components. There is also intrinsic internal structure among the components, such as the affinity between each pair. When reducing the dimensionality of GMM components, it is necessary to maintain these two types of structure. Motivated by the idea of linear discriminative analysis [23] and Maximum Margin Criterion [24], DCAR aims to minimize the intra-class distance while simultaneously maximizing the inter-class distance. In the next subsection, we introduce an undirected graph defined by a real symmetric affinity matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ that encodes these structures.

2) Affinity Matrix Construction: The affinity matrix $\mathbf{A}$ is defined by building an intra- (within-)class similarity graph and an inter- (between-)class similarity graph, as follows:

$$\mathbf{A} = \mathbf{S}_w - \mathbf{S}_b. \tag{5}$$

$\mathbf{S}_w$ and $\mathbf{S}_b$ are two binary matrices describing the intra-class and inter-class similarity graphs respectively, formulated as:

$$\mathbf{S}_w(i,j) = \begin{cases} 1 & \text{if } g_i \in \text{NN}_w(g_j) \text{ or } g_j \in \text{NN}_w(g_i) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

and

$$\mathbf{S}_b(i,j) = \begin{cases} 1 & \text{if } g_i \in \text{NN}_b(g_j) \text{ or } g_j \in \text{NN}_b(g_i) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $\text{NN}_w(g_i)$ contains the n_w nearest neighbors of component g_i that share the same label as ℓ_i, and $\text{NN}_b(g_i)$ is the set of n_b nearest neighbors of g_i that have different labels. Here, the nearest neighbors of each component can be identified via their similarity. We use heat kernel weight to measure the similarity between components

$$\boldsymbol{S}(i,j) = \lambda \exp\left(\frac{-\delta_\mu^2(\mu_i,\mu_j)}{2\sigma_\mu^2}\right) + \exp\left(\frac{-\delta_\Sigma^2(\Sigma_i,\Sigma_j)}{2\sigma_\Sigma^2}\right) \tag{8}$$

where λ is a trade-off parameter to control the contribution from the components' means and covariance matrices and δ_μ indicates the distance measure for the means of the mixture components. Here we use the simple euclidean distance

$$\delta_\mu^2(\mu_i,\mu_j) = \|\mu_i - \mu_j\|_2^2. \tag{9}$$

δ_Σ indicates the distance measure for the covariance matrices of the components.

Previous research uses a number of metrics, including the Affine-Invariant Riemannian Metric (AIRM) [25], Stein Divergence [26], and the Log-Euclidean Metric (LEM) [27]. AIRM imposes a high computational burden in practice, and we have observed experimentally that nearest neighbors selected according to LEM more often fall into the same event than nearest neighbors selected according to either AIRM or Stein. For these reasons, we use LEM to compute δ_Σ

$$\delta_\Sigma^2(\Sigma_i,\Sigma_j) = \|\log(\Sigma_i) - \log(\Sigma_j)\|_F^2 \tag{10}$$

where the log(.) function is the matrix logarithm.

The constructed affinity matrix $\mathbf{A}$ thus effectively combines local structure, i.e., nearest neighbors, and global structure, i.e., label information—which is used to find the within-class nearest neighbor (NN_w) and the between-class nearest neighbor (NN_b).

3) Embedding Optimization: Once we have $\mathbf{A}$, the next step is to learn an embedding such that the structure among the original GMM components $\{g_i\}_{i=1}^N = \{\mu_i, \Sigma_i\}_{i=1}^N$ is reflected by the low-dimensional mixture components $\{\hat{g}_i\}_{i=1}^N = \{\hat{\mu}_i, \hat{\Sigma}_i\}_{i=1}^N$. This process can be modeled using the following optimization problem:

$$\begin{aligned} F(\mathbf{W}) &= \sum_{i,j} A_{ij}\left(\lambda\delta_\mu^2(\hat{\mu}_i,\hat{\mu}_j) + \delta_\Sigma^2(\hat{\Sigma}_i,\hat{\Sigma}_j)\right) \\ &= \sum_{i,j} A_{ij}\Big(\lambda\|\mathbf{W}^T(\mu_i-\mu_j)\|_F^2 \\ &\quad + \|\log(\mathbf{W}^T\Sigma_i\mathbf{W}) - \log(\mathbf{W}^T\Sigma_j\mathbf{W})\|_F^2\Big). \end{aligned} \tag{11}$$

With the aid of the mapping functions in (3) and (4) and the distance metrics δ_μ (9) and δ_Σ (10), the optimization problem can be rewritten as

$$\min_{\mathbf{W}^T\mathbf{W}=I_r} F(\boldsymbol{W}). \tag{12}$$

As in (8), λ is used to balance the effects of two terms, in tuning by cross-validation on the training data. Optimizing $\mathbf{F}(\mathbf{W})$ results in a situation where the low-dimensional components are close if their corresponding original high-dimensional components are event-aware neighbors; otherwise, they will be as far apart as possible.

The problem in (12) is a typical optimization problem with orthogonality constraints, which can be effectively solved on Grassmannian manifolds [28]. For $\mathbf{F}(\mathbf{W})$ and any rotation matrix $\mathbf{R} \in SO(r)$ (i.e., $\mathbf{R}\mathbf{R}^T = \mathbf{R}^T\mathbf{R} = \mathbf{I}_r$), we have $\mathbf{F}(\mathbf{W}) = \mathbf{F}(\mathbf{W}\mathbf{R})$ (see Appendix A for a detailed proof). This indicates that (12) is most compatible with a Grassmannian manifold. In other words, we can model the embedding $\mathbf{W}$ as a point on a Grassmannian manifold $\mathcal{G}(r, d)$, which consists of the set of all linear r-dimensional subspaces of $\mathbb{R}^d$.

Here we employ the conjugate gradient (CG) technique to solve (12), because CG is easy to implement, has low storage requirements, and provides superlunar convergence in the limit [28]. On a Grassmannian manifold, CG performs minimization along geodesics with specific search directions. Here, the geodesic is the shortest path between two points on the manifold. For every point on the manifold $\mathcal{G}$, its tangent space is a vector space that contains the tangent vectors of all possible curves passing through that point. Unlike flat spaces, on a manifold we cannot directly transport a tangent vector from one point to another point by simple translation; rather, the tangent vectors must parallel transport along the geodesics. (Details for finding the optimal $\mathbf{W}$ are given in Appendix B.)

Then, given a new audio track, we can extract its MFCC features, train P GMM components, and re-represent these components with the embedding $\boldsymbol{W}$ to get its discriminative, low-dimensional mixture components, i.e., its DCAR.

IV. Classification and Detection With DCARs

As we describe above, each audio track is represented via several mixture components, including mean vectors and covariance matrices. It would be possible to flatten the matrices into vectors and then use traditional vector-based classification methods for event detection. However, the covariance matrices lie on the manifold of positive definite matrices, and such a vectorization process would ignore this manifold structure [18]. Therefore, we use the Kernel Ridge Regression (KRR) method to build the event and scene classifiers.

Let $\hat{G} = \{\hat{g}_i\}_{i=1}^N$ and $\hat{g}_i = \{\hat{\mu}_i, \hat{\Sigma}_i\}$ be mixture components for the training audio tracks belonging to L events. (In order to reduce the computational complexity, following [16], we can generate components for each class instead of for each track.) $Y \in \mathbb{R}^{N\times L}$ indicates the label information for $\hat{G}$, where $\mathbf{Y}_{ij} = 1$ if the ith component belongs to the jth event; otherwise $\mathbf{Y}_{ij} = 0$. The KRR method aims to train a classifier by solving the following optimization problem:

$$\min_{\mathbf{H}} \mathbf{J}(\mathbf{H}) = \|\phi(\hat{G})^T\mathbf{H} - \mathbf{Y}\|_F^2 + \alpha\|\mathbf{H}\|_F^2 \qquad (13)$$

where ϕ is a feature mapping from the original feature space to a high-dimensional space, and the kernel function can be written as $K = \phi(\hat{G})^T\phi(\hat{G})$.

Since each component $\hat{g}_i$ has a mean $\hat{\mu}_i$ and a covariance matrix $\hat{\Sigma}_i$, we can define a combined kernel function to integrate these two parts, as follows:

$$K(\hat{g}_i, \hat{g}_j) = \lambda K_\mu(\hat{\mu}_i, \hat{\mu}_j) + K_\Sigma(\hat{\Sigma}_i, \hat{\Sigma}_j). \qquad (14)$$

The trade-off parameter λ (see (8)) can be tuned by cross-validation on the training data. As described in Section III-B2, we use a Gaussian kernel to calculate K_μ and K_Σ via

$$K_\mu(\hat{\mu}_i, \hat{\mu}_j) = \exp\left(\frac{-\|\hat{\mu}_i - \hat{\mu}_j\|_2^2}{2\sigma_{\hat{\mu}}^2}\right) \qquad (15)$$

and

$$K_\Sigma(\hat{\Sigma}_i, \hat{\Sigma}_j) = \exp\left(\frac{-\|\log(\hat{\Sigma}_i) - \log(\hat{\Sigma}_j)\|_F^2}{2\sigma_{\hat{\Sigma}}^2}\right). \qquad (16)$$

The problem in (13), as a quadratic convex problem, can be optimized by setting its derivative with respect to H to zero, and then computing H in closed form

$$\mathbf{H} = \phi(\hat{G})(K + \alpha\mathbf{I})^{-1}\mathbf{Y}. \qquad (17)$$

Here $K_{ij} = K(\hat{g}_i, \hat{g}_j)$ as given in (14).

Given a new test audio track, P mixture components $\{g_p\}_{p=1}^P = \{w_p, \mu_p, \Sigma_p\}_{p=1}^P$ can be obtained via the methods described in Section III-A. Then the corresponding discriminative, low-dimensional mixture components $\{\hat{g}_p\}_{p=1}^P$ can be generated as in (3) for $\hat{\mu}_p = \boldsymbol{W}^T\mu_p$, and as in (4) for $\hat{\Sigma}_p = \boldsymbol{W}^T\Sigma_p\boldsymbol{W}$, where the embedding $\mathbf{W}$ is learned from the training data. Then the class membership matrix $M = \{M_p\}_{p=1}^P$ (where $M_p \in \mathbb{R}^{1\times L}$ is the class membership of the p-th component) can be calculated

$$\begin{aligned} M_p &= \phi(\hat{g}_p)^T\mathbf{H} = \phi(\hat{g}_p)^T\phi(\hat{G})(K + \alpha\mathbf{I})^{-1}\mathbf{Y} \\ &= K_p(K + \alpha\mathbf{I})^{-1}\mathbf{Y}. \end{aligned} \qquad (18)$$

Here $K_p = [K(\hat{g}_p, \hat{g}_i)]_{i=1}^N$, indicating the similarity between $\hat{g}_p$ and all of the training mixture components in $\hat{G}$. We can then make a final prediction about the class of the new audio track with P components using an average voting scheme

$$\ell = \arg\max_j \sum_{p=1}^P w_p M_p(j) \qquad (19)$$

where w_p is the weight of the pth component. The computational complexity for embedding learning and classification processes is analyzed in Appendix C.

V. Event Detection Experiments on YLI-MED

We evaluated the event detection and classification performance of our proposed representation against several baseline representations, using the recently released dataset YLI-MED.

A. Dataset

YLI-MED [29] is an open video corpus focused on *multimedia* event detection research (modeled on the TRECVID MED corpus [30], but publicly available). The videos are drawn from the YFCC100M dataset [31]. YLI-MED includes about 2000 videos that contain examples of ten events, with standard training and test sets. Our experiments used the audio tracks, which we preprocessed and represented using MFCCs. The frames were extracted using a 100 ms Hamming window with a stride size of 10 ms per frame shift, a 22050 Hz sampling rate, and

TABLE I
YLI-MED DATASET COMPOSITION

Event ID	Event Name
Ev101	Birthday Party
Ev102	Flash Mob
Ev103	Getting a Vehicle Unstuck
Ev104	Parade
Ev105	Person Attempting a Board Trick
Ev106	Person Grooming an Animal
Ev107	Person Hand-Feeding an Animal
Ev108	Person Landing a Fish
Ev109	Wedding Ceremony
Ev110	Working on a Woodworking Project

40 Mel bands. The first 20 coefficients and the corresponding ∇ and acceleration coefficients ($\nabla\nabla$) were calculated to obtain a 60-dimensional vector for each frame. Table I describes the data we used. The number of videos in each class varies from 89 to 99 for training, and 39 to 131 for testing.

B. Methodology

To evaluate our proposed DCAR method, we compared it with four state-of-the-art audio representations used for event detection and classification: mv-vector [15], i-vector [14], GMM, and HEM-GMM [16]. By GMM, we mean here the base GMMs obtained by extracting the GMM components from each audio track (as described in Section III-A), but without the discriminative dimensional reduction step. For comparison, we also included the DCAR(c) representation, which is similar to HEM-GMM in that it generates a few components for each class (rather than each track).

As we mentioned in Section II, an i-vector obtains a GMM supervector for each frame, then factorizes them to obtain the i-vector representation. In contrast, an mv-vector models each audio track via the mean and variance of the MFCC features, then concatenates the mean and variance.

There are several parameters for each of the representations, which we tuned using cross-validation on the training data to obtain the best result. For GMM, HEM-GMM, DCAR, and DCAR(c), the number of components for each audio track is tuned from 1 to 10, with a step of 1. For i-vector, the number of components in all of the training data is tuned to one of the values in $\{2^7, 2^8, 2^9, 2^{10}\}$ and the vector dimensionality is tuned to one of the values in $\{200, 400, 600, 800, 1000\}$. For HEM-GMM and DCAR(c), the number of components for each class is tuned form 50 to 200, with a step of 50. For DCAR and DCAR(c), the number of nearest neighbors (n_w and n_b) is set to be 5 for affinity matrix construction, the embedding space size r is tuned in $[L, 60]$ with a step of 5 (L is the number of events), and the trade-off parameter λ is tuned to one of $\{10^k\}_{k=-2}^{2}$.[2]

[2]In addition, we tried normalizing each track, with each MFCC feature having a mean of 0 and a variance of 1. All GMM components then have 0 means, so KRR depends solely on the covariance matrix. However, for event detection, information about means appears to be important. When we tuned λ in (14), we found that we usually obtained the best results with both means and covariance matrices.

Because our focus is on comparing different audio representations, we describe here experiments that all used the same classification method, KRR.[3] When applying KRR, we use the Gaussian kernel (15) for i-vector and mv-vector, and the combined kernel (14) for the other methods. The parameters (σ_μ and σ_Σ) in the heat kernel weight are set by a self-tuning technique [32]. For the l-th event in s testing tracks, we compared the prediction result to the ground truth to determine the number of true positives (TP_l), false positives (FP_l), true negatives (TN_l), and false negatives (FN_l). We then evaluated event detection or classification performance using four metrics, $Accuracy$, $FScore$, False Alarm Rate (FAR), and $MissRate$, defined (respectively) as

$$Accuracy = \frac{\sum_{l=1}^{L} TP_l}{s}, FScore_l = \frac{2 \times TP_l}{2 \times TP_l + FP_l + FN_l}$$
$$FAR_l = \frac{FP_l}{TN_l + FP_l}, \quad MissRate_l = \frac{FN_l}{FN_l + TP_l}.$$

$Accuracy$ is calculated on all s testing tracks together (i.e., the combined or overall accuracy), while the other three metrics are calculated for each event and then averaged for the L events. Larger $Accuracy$ and $FScore$ values and smaller FAR and $MissRate$ values indicate better performance.

VI. Experimental Results: Event Detection

We evaluated the representations under study in a combination of binary detection and multi-event classification tasks.

A. Binary-Event Detection

In the first experiment, we built 45 binary classifiers (one for each pair of events). We had two goals in this experiment. The first was to compare two representation strategies: modeling GMMs on all training tracks, in this case as the first phase of the i-vector approach, vs. modeling GMMs on each training track, using DCAR. The second was to investigate *how* the events are distinguished. As the graphs in Fig. 2 show, DCAR outperforms i-vector on most tasks. On average, DCAR achieves an accuracy improvement of 10.74% over i-vector (0.8293 vs. 0.7489) across the binary detection tasks. The win-tie-loss value for pairwise tests with 0.05 accuracy-difference threshold for DCAR against i-vector is 35-7-3; with 0.01 difference threshold, it is 40-2-3. Note that, when the absolute accuracy difference is less than the threshold, the situation is called a tie.

From these results, we can see that there are some event pairs that are particularly difficult to distinguish, such as Ev106–Ev107 and Ev107–Ev108. And in fact, it could be argued that distinguishing between events with a *Person Grooming an Animal* (Ev106), a *Person Hand-Feeding an Animal* (Ev107), and a *Person Landing a Fish* (Ev108) can be non-trivial even for humans. Nonetheless, compared with i-vector, our proposed DCAR increases binary classification accuracy even on these

[3]To check the validity of this approach, we also tested several other classification techniques with mv-vector and i-vector representations, including SVM, KNN, and PLDA. The performance rankings between representations were parallel across all classification techniques.

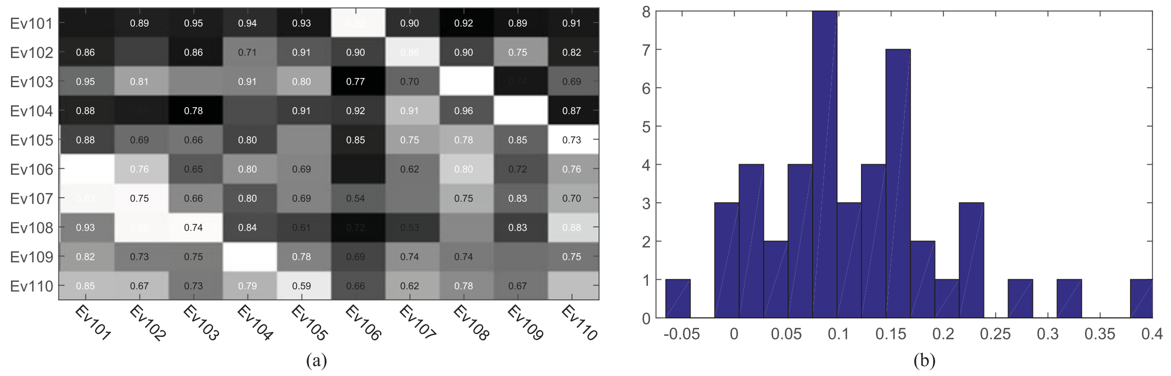


Fig. 2. Binary classification accuracy. (a) Comparison between i-vector (in the lower left triangle) and DCAR (in the upper right triangle) for each pairwise classification, with darker color indicating higher accuracy. (b) Histogram of the accuracy improvement obtained by DCAR relative to i-vector across the 45 classifications. (a) I-vector (lower triangle) versus DCAR (upper triangle). (b) Histogram of relative gain (DCAR versus i-vector).

difficult pairs. This result demonstrates that modeling each audio track via a Gaussian mixture model is more suitable to characterizing audio content, and that integrating label information and local structure is useful in generating discriminative representations.

B. Easy Versus Hard Cases

To further explore how our proposed DCAR method performs on audio-based event detection under different difficulty levels, we extracted two subsets from YLI-MED. Subset EC5 (“EasyCase”) contains five events (Ev101, Ev104, Ev105, Ev108, and Ev109) that are generally easy to distinguish from (most of) the others. Subset HC4 (“HardCase”) contains four events (Ev103, Ev106, Ev107, and Ev110) that are more difficult to distinguish.[4]

1) Dimensionality Tuning: Before comparing DCAR with other state-of-the-art representations, we conducted a set of multi-event classification experiments to study how the dimensionality parameter (r) affects DCAR under these two difficulty levels. Here, we used five-fold cross-validation on the training data to tune r. The parameter was tuned from 5 to 60 with a step of 5. Combined (overall) $Accuracy$ and average $MissRate$ on EC5 and HC4 for each step are given in Fig. 3.

The results show that DCAR performs better as r increases, reaches the best value at $r = 25$ for both cases, and then again decreases in performance as r grows larger. We believe this is because a smaller r cannot optimally characterize the hidden structure of the data, while a larger r may separate the structure into more dimensions until it is essentially equivalent to the original data, thus decreasing the efficacy of the representation.

2) Easy and Hard Results: Table II shows the multi-event classification performance of DCAR, DCAR(c), and four baseline representations—base GMM, HEM-GMM, mv-vector, and i-vector—in terms of $FScore$, $Accuracy$, $MissRate$ and FAR at the two difficulty levels (on EC5 and HC4).

For the EC5 subset, DCAR(c) consistently achieves the best results (marked in bold) on each evaluation metric, with DCAR consistently second-best. For HC4, the situation is nearly reversed, with DCAR achieving the best results on most metrics. However, differences between DCAR and DCAR(c) were not statistically significant in either case. (Significance is assessed using McNemar’s two-tailed test for correlated proportions). Interestingly, i-vector performs better than mv-vector on the EC5 data, but worse than mv-vector on the HC4 data ($p = 0.0001$ or better for $Accuracy$ in both cases). For $Accuracy$, $p = 0.01$ or better on pairwise comparisons of DCAR vs. mv-vector and i-vector, and $p = 0.05$ or better for DCAR vs. GMM only, for both subsets. Comparing DCAR(c) with HEM-GMM, $p = 0.002$ on EC5, but $p = 1$ on HC4.

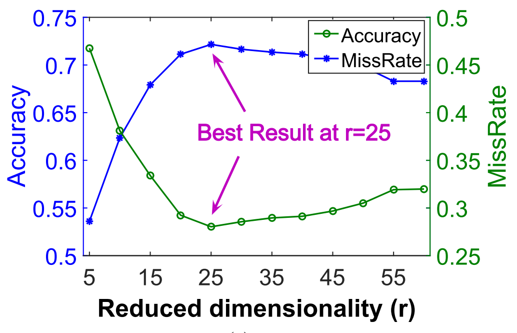
3) DCAR, Variance, and Structure: We can make a number of observations about these results. First, it seems that modeling a GMM on all the training audio tracks together (as in i-vector) is not effective comparing with modeling GMM components for each audio track when the events are semantically related to each other (as in HC4). We believe this is due to the fact that, in real-world applications (e.g., with user-generated content), each audio track may have a large variance. The set of strategies that model each track via GMM capture the hidden structure within each audio track, while the i-vector strategy may smooth away that structure (even between events), leading to a less useful representation.

Second, mv-vector performs worse than the proposed methods. We believe this indicates that one mixture component may not sufficiently capture the full structure of the audio; in addition, vectorizing the mean and variance inevitably distorts the intrinsic geometrical structure among the data. Third, the proposed methods outperform GMM and HEM-GMM. As we described in Section III, DCAR begins by extracting such a GMM model, but it also takes into account the label information and the intrinsic nearest neighbor structure among the audio tracks when modeling the training data, and outputs a mapping function to effectively represent the test data.

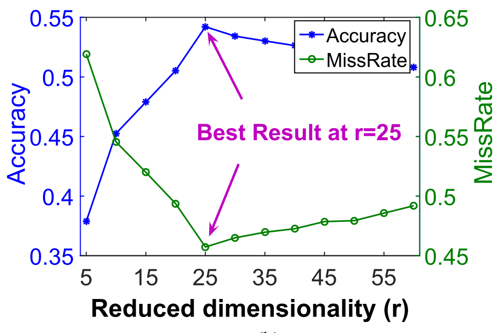
C. Ten-Event Classification

Because event classification is a kind of supervised learning, learning becomes more difficult as the number of events increases. In the next experiment, we again compared the proposed DCAR model with the baseline representations, this time on the ten-event dataset. As before, the parameters for each

[4]The division was based on the results in Sections VI-A and VI-C, as well as *a priori* human understanding of the events’ similarity. Because multiple criteria were used, Ev102 did not fall clearly into either category.



(a)



(b)

Fig. 3. Effects of varying the parameter (r) in DCAR for the EasyCase subset (five-event classification), and for the HardCase subset (four-event classification), in terms of $Accuracy$ and $MissRate$. (a) EC5. (b) HC4.

TABLE II
COMPARISON OF SIX REPRESENTATIONS [MV-VECTOR, I-VECTOR, GMM, HEM-GMM, DCAR, AND DCAR(C)] ON MULTIEVENT CLASSIFICATION FOR EASY-TO-DISTINGUISH EVENTS (EC5) AND HARD-TO-DISTINGUISH EVENTS (HC4)

Evaluation	Subset EC5						Subset HC4					
	(Ev101,Ev104, Ev105, Ev108, Ev109)						(Ev103,Ev106, Ev107, Ev110)					
Metric	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)
FScore(↑)	0.4773	0.6415	0.6670	0.6466	<u>0.7067</u>	**0.7164**	0.4278	0.2795	0.4821	0.5059	**0.5282**	<u>0.5261</u>
Accuracy(↑)	0.5455	0.6828	0.7131	0.6949	<u>0.7434</u>	**0.7475**	0.4573	0.2863	0.5000	<u>0.5531</u>	**0.5684**	0.5513
MissRate(↓)	0.5168	0.3367	0.3252	0.3486	<u>0.2779</u>	**0.2710**	0.5393	0.6975	0.4840	0.4770	<u>0.4577</u>	**0.4536**
FAR(↓)	0.1136	0.0785	0.0730	0.0771	<u>0.0647</u>	**0.0640**	0.1788	0.2409	0.1684	**0.1496**	**0.1496**	0.1564

Note: Best results in boldface; second-best underlined.

TABLE III
PER-EVENT COMPARISON OF CLASSIFICATION PERFORMANCE (AS $FScore$ AND $MissRate$) USING SIX REPRESENTATIONS: MV-VECTOR, I-VECTOR, GMM, HEM-GMM, DCAR, AND DCAR(C)

	FScore (↑)						MissRate (↓)					
	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)
Ev101	0.7259	<u>0.7842</u>	0.7303	0.7518	0.7835	**0.7904**	0.2824	0.1679	0.1527	0.1908	<u>0.1298</u>	**0.1221**
Ev102	0.2837	0.3396	0.3651	0.3529	**0.4603**	<u>0.4426</u>	0.5918	0.6327	0.5306	0.5102	**0.4082**	<u>0.4490</u>
Ev103	0.2178	0.2569	0.2410	0.3639	<u>0.3820</u>	**0.4138**	0.7179	0.6410	0.7436	0.6154	<u>0.5641</u>	**0.5385**
Ev104	0.4274	0.6206	0.6000	0.5820	<u>0.6207</u>	**0.6421**	0.6063	0.4331	0.3622	0.4409	<u>0.3621</u>	**0.3150**
Ev105	0.3354	0.3899	**0.5714**	<u>0.5424</u>	0.5178	0.5419	0.6932	0.6477	<u>0.3864</u>	0.4545	0.4205	**0.3750**
Ev106	0.1964	0.1835	0.2963	0.2162	<u>0.3750</u>	**0.3846**	0.7105	0.7368	<u>0.6842</u>	0.7895	**0.6053**	**0.6053**
Ev107	0.3850	0.3298	0.3250	**0.4409**	0.4024	<u>0.4096</u>	<u>0.6814</u>	0.7257	0.7699	**0.6372**	0.7080	0.6991
Ev108	0.3191	0.3853	0.3878	0.3186	<u>0.4231</u>	**0.4536**	0.6341	<u>0.4878</u>	0.5366	0.5610	**0.4634**	**0.4634**
Ev109	0.4211	0.5028	0.4286	0.4894	<u>0.5176</u>	**0.5263**	0.6667	<u>0.5833</u>	0.6667	**0.5741**	0.5926	<u>0.5833</u>
Ev110	0.0833	0.2299	**0.2857**	<u>0.2703</u>	0.2162	0.2571	0.9091	**0.7727**	<u>0.7500</u>	**0.7727**	0.8182	0.7955
Average	0.3395	0.4023	0.4231	0.4330	<u>0.4699</u>	**0.4862**	0.6494	0.5829	0.5583	0.5546	<u>0.5072</u>	**0.4946**

Note: Best results in boldface; second-best underlined.

method were tuned by cross-validation on the training data. Table III gives the classification performance on each event and the average over the ten events in terms of $FScore$ and $MissRate$.

For all of the individual events and on average, DCAR achieves superior or competitive performance. DCAR also performs better in terms of $Accuracy$ and FAR. The overall $Accuracy$ scores for mv-vector, i-vector, GMM, HEM-GMM, DCAR, and DCAR(c) are 0.3907, 0.4640, 0.4923, 0.4859, 0.5231, and **0.5488**, respectively ($p = 0.02$ or better for DCAR vs. each baseline and $p = 0.0002$ or better for DCAR(c) vs. each baseline [McNemar's two-tailed]). The average FAR scores are 0.0674, 0.0593, 0.0570, 0.0571, 0.0523, and **0.0506**. Although other representations may perform as well or better on some particular events, DCAR and DCAR(c) consistently outperform the other representations for all evaluation metrics (an average of more than 8% gain on all metrics relative to the second best representation). These results further demonstrate that modeling each audio track via GMM and then integrating both label information and local structure are beneficial.

In addition to comparing results across the existing representation methods, we also experimented with applying feature

TABLE IV
COMPARISON OF TEN-EVENT CLASSIFICATION PERFORMANCE FOR GMM REPRESENTATIONS WITH AND WITHOUT PRE-TRAINING FEATURE REDUCTION

Evaluation	GMM	PCA+GMM	LDA+GMM
Metric		$r = 30$	$r = 9$
FScore (↑)	0.4231	0.4293	0.3278
Accuracy (↑)	0.4923	0.4987	0.3419
MissRate (↓)	0.5583	0.5508	0.6574
FAR (↓)	0.0570	0.0562	0.0935

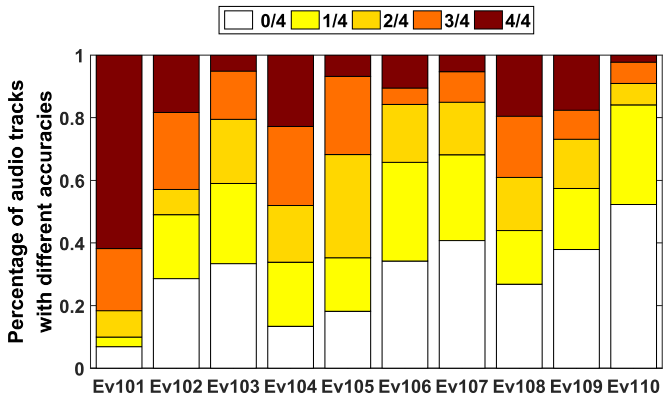


Fig. 4. Per-event percentages of test tracks that are correctly classified by how many representations.

reduction methods at the frame level before training the GMM, using PCA [33] and linear discriminant analysis (LDA) [23]. (As an alternative to DCAR's approach to dimensionality reduction.) The number of principal components (r) in PCA was tuned from 5 to 60 with a step of 5. For LDA, $r = L - 1$, where L is the number of events. As shown in Table IV, the results with PCA are a little better than GMM without PCA, but the accuracy difference is not statistically significant (p = 0.7117, McNemar's two-tailed). Results with LDA are much worse than GMM without LDA. We hypothesize that the main reason for the poor performance of LDA+GMM is that LDA only considers $L - 1$ dimensions, which is usually too few to capture sufficient information for later GMM training.

D. Intra-Event Variation

Delving deeper into the variability of audio tracks, we looked at the degree to which some test tracks in YLI-MED could be classified more accurately than others.

We took the predicted result for each test track in the experiments with four of the representations (mv-vector, i-vector, GMM and DCAR) described in Section VI-C and calculated how many of the representations made correct predictions for that track.

Fig. 4 shows the distribution of the number of representations making accurate predictions for each audio track, broken down by event. Generally, there is wide variation in accuracy among audio tracks belonging to the same event, with the exception of Ev101 (Birthday Party); this suggests that Ev101 may have distinctive audio characteristics that lead to more consistent classification. It is worth noting that there are some audio tracks that

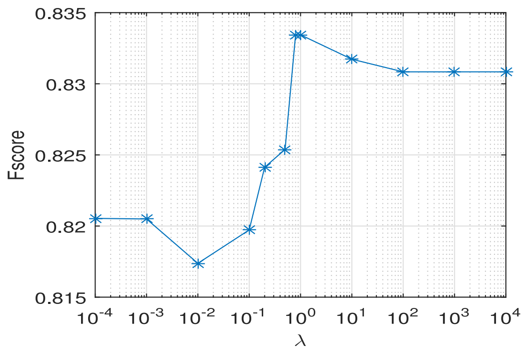
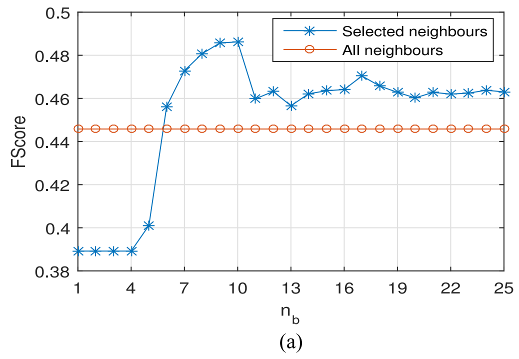


Fig. 5. Effects of λ on DCAR for DCASE.



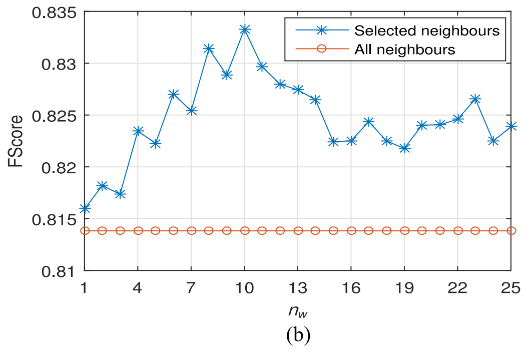


Fig. 6. Effects of the number of nearest neighbors on DCAR for DCASE. (a) $n_w = 5$. (b) $n_b = 5$.

are never correctly classified by any of the representations. For example, more than 50% of the audio tracks for Ev110 (Working on a Woodworking Project) could not be correctly classified by any of the four representations. This situation highlights a challenging property of the event detection task: some of the events are quite confusable due to their inherent characteristics. For example, Ev103 (Getting a Vehicle Unstuck) may have similar audio properties to Ev110 (Working on a Woodworking Project) in that both are likely to involve sounds generated by motors. This is also the reason we included Ev103 and Ev110 in the "Hard Case" HC4 dataset for the experiments described in Section VI-B.

VII. SCENE CLASSIFICATION EXPERIMENTS ON DCASE

Extending beyond event detection, we evaluated the performance of our proposed representation on acoustic scene classification, defined as recognition of the audio environment.

TABLE V
PER-FOLD COMPARISON OF ACOUSTIC SCENE CLASSIFICATION PERFORMANCE USING SEVEN METHODS: DCASE'S BASE, MV-VECTOR, I-VECTOR, GMM, HEM-GMM, DCAR, AND DCAR(C), BASED ON MONAURAL MFCC FEATURES

	FScore (↑)							MissRate (↓)						
	Base	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)	Base	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)
Fold1	0.7197	0.7100	0.8020	0.8273	0.8230	**0.8383**	0.8152	0.2685	0.2750	0.1979	0.1655	0.1751	**0.1573**	0.1893
Fold2	0.7541	0.5845	0.6650	0.7619	0.7766	0.7823	**0.7882**	0.2413	0.3981	0.3196	0.2294	0.2222	0.2138	**0.2061**
Fold3	0.6135	0.7512	0.7684	0.8149	0.7990	0.8283	**0.8320**	0.3699	0.2370	0.2172	0.1748	0.1921	0.1639	**0.1609**
Fold4	0.7089	0.7053	0.7691	0.8032	0.7930	**0.8025**	0.7991	0.2677	0.2778	0.2230	**0.1915**	0.2041	0.1927	0.1959
Average	0.6990	0.6877	0.7511	0.8018	0.7979	**0.8129**	0.8086	0.2868	0.2969	0.2394	0.1903	0.1977	**0.1819**	0.1881
	Accuracy (↑)							FAR (↓)						
Average	0.7132	0.7031	0.7606	0.8097	0.8026	**0.8181**	0.8119	0.0205	0.0210	0.0172	0.0136	0.0141	**0.0130**	0.0134

Note: Best results in boldface; second-best underlined.

TABLE VI
PER-FOLD COMPARISON OF ACOUSTIC SCENE CLASSIFICATION PERFORMANCE USING SEVEN METHODS: DCASE'S BASE, MV-VECTOR, I-VECTOR, GMM, HEM-GMM, DCAR, AND DCAR(C), BASED ON BINAURAL MFCC FEATURES

	FScore (↑)							MissRate (↓)						
	Base	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)	Base	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)
Fold1	0.7123	0.7491	0.8195	0.8310	0.8286	0.8392	**0.8422**	0.2795	0.2426	0.1765	0.1647	0.1706	**0.1569**	0.1603
Fold2	0.7540	0.6166	0.6899	0.7871	0.8036	0.8065	**0.8222**	0.2516	0.3730	0.2928	0.2065	0.1946	0.1893	**0.1728**
Fold3	0.6580	0.7583	0.7792	0.8222	0.8384	0.8313	**0.8402**	0.3275	0.2324	0.2162	0.1671	0.1554	0.1600	**0.1496**
Fold4	0.7215	0.7909	0.8273	0.8279	0.8183	**0.8522**	0.8206	0.2650	0.2022	0.1660	0.1680	0.1817	**0.1451**	0.1654
Average	0.7115	0.7287	0.7790	0.8170	0.8210	0.8323	**0.8333**	0.2809	0.2626	0.2128	0.1766	0.1756	0.1628	**0.1620**
	Accuracy (↑)							FAR (↓)						
Average	0.7191	0.7374	0.7871	0.8234	0.8244	0.8372	**0.8380**	0.0201	0.0188	0.0153	0.0126	0.0125	0.0117	**0.0116**

Note: Best results in boldface; second-best underlined.

A. Dataset

We used the TUT Acoustic Scenes 2016 dataset from the DCASE challenge [34]. This dataset is divided into 15 classes of scene from three scenes (Vehicle, Outdoor, Indoor), with (in most cases) several 3–5 minute binaural recordings from different locations for each class. Since the ground truth labels for the evaluation set were not public when we conducted our experiments, we used only the development set. The development set is divided into 4 folds, each of which has a training and a test set (prescribed by DCASE). We experimented with each fold separately, then recorded the performance on each fold. The 5-fold cross-validation on each fold was used to tune the parameters, i.e., the training files of each fold are partitioned into five equal subsets.

For each scene, the development set includes 78 segments of 30 s each. We extracted MFCCs from the DCASE segment files using similar settings to those for YLI-MED, except we extracted the frames using a 40 ms Hamming window with 50% overlap [34].

B. Methodology

The classifier (KRR), parameters, and evaluation metrics we used for audio scene classification parallel those we used for event classification, described in Section V-B. In addition to comparing DCAR and DCAR(c) with our mv-vector, i-vector, GMM, and HEM-GMM baselines, we also included a baseline provided by DCASE (which we will refer to as *Base*). For Base, the Gaussian mixture components are learned for each acoustic scene using the Expectation-Maximization algorithm.

C. Experimental Results: Scene Classification

First, we investigated the effect of the trade-off parameter λ on DCAR. The chart in Fig. 5 shows that the best result is obtained at around $\lambda = 1$; this indicates that both mean and covariance are useful for determining the low-dimensional components. Meanwhile, we compared the performance of DCAR using only the mean vector vs. only the covariance matrix, obtaining an $FScore$ of 0.7279 and 0.7473 respectively – notably worse than DCAR using both (0.8129).[5] We also investigated the effect of the number of nearest neighbors when constructing the affinity matrix. As shown in Fig. 6, the performance of DCAR increases with the number of neighbors, approaches a maximum, then decreases.[6] In other words, a large number of neighbors may bring noisy information. (Red indicates performance without considering local structure, i.e., with each element of $\boldsymbol{A}$ set to 1.) These explorations confirm that DCAR obtains higher performance by using both local structure and global label information.

We then tested the various methods in both monaural and binaural conditions. Table V gives the classification performance

[5]The same comparison on YLI-MED data shows the opposite balance between mean vector and covariance matrix, but again, the best result is obtained by using both.

[6]The same pattern is found with the YLI-MED data.

TABLE VII
PER-CLASS COMPARISON OF AVERAGE ACOUSTIC SCENE CLASSIFICATION $Accuracy$ USING SEVEN METHODS: DCASE'S BASE, MV-VECTOR, I-VECTOR, GMM, HEM-GMM, DCAR, AND DCAR(C), BASED ON BINAURAL MFCCS

Group	Scene	Base	mv-vector	i-vector	GMM	HEM-GMM	DCAR	DCAR(c)
Vehicle	Bus	0.6197	0.7204	0.6816	0.7849	0.8947	0.7836	**0.9079**
Vehicle	Car	0.6888	0.8467	**0.9480**	0.8974	0.7711	0.8599	0.8217
Vehicle	Train	0.3913	0.4408	0.6477	0.5505	**0.8760**	0.5745	**0.9028**
Vehicle	Tram	0.9205	0.8397	0.8561	0.8674	0.8599	0.8813	**0.9375**
Indoor	Cafe	0.4963	0.7376	0.7050	0.8986	**0.9378**	0.8766	0.9360
Indoor	Grocery store	0.5959	0.6930	0.8208	0.6692	**0.9722**	0.7719	**0.9722**
Indoor	Home	**0.8222**	0.6826	0.7667	0.7722	0.6692	0.7056	0.7074
Indoor	Library	0.8279	0.5569	0.7093	0.8171	0.7497	**0.8929**	0.6931
Indoor	Office	**0.9737**	0.7139	**0.9737**	0.8238	0.8790	0.8407	0.9087
Indoor	Metro station	**1.0000**	0.9452	0.9591	0.9737	0.9737	0.9737	0.9459
Outdoor	City center	0.8819	0.8756	0.9097	0.9247	**0.9733**	0.9090	0.8794
Outdoor	Forest path	0.7242	0.8710	0.8770	**0.9861**	0.7125	**0.9861**	0.7042
Outdoor	Urban park	0.2861	0.5417	0.4458	0.6889	**0.7707**	0.7667	0.7600
Outdoor	Lakeside beach	0.6930	0.8327	0.8816	0.8948	0.6109	**0.9079**	0.6424
Outdoor	Residential area	**0.8653**	0.7638	0.6253	0.8020	0.8510	0.8271	0.8510
	Avg. of classes	0.7191	0.7374	0.7871	0.8234	0.8244	0.8372	**0.8380**
	Variance	0.0447	0.0198	0.0225	0.0142	0.0253	**0.0112**	0.0244

Note: Best results in boldface; second-best underlined.

for the different representations (as $FScore$ and $MissRate$), across the four folds and on average, based on monaural MFCCs. Again, the mv-vector method computes the mean and variance of the frames' MFCCs for each audio file, GMM and DCAR learn the file-aware components, HEM-GMM and DCAR(c) learn the class-aware components, and i-vector learns the components from all of the files, while Base learns the components from the audio files belonging to one scene. In most cases, DCAR obtains a better result (shown in boldface) than the baselines on these metrics.

The best performance in DCASE 2016 was obtained by Eghbal-Zadeh *et al.* [35]. In their work, binaural audio (from both left and right channels), the averaged monaural representation, and the difference between the left and right channels were extracted as four different feature-space representations for each audio track (using 60-dimensional MFCCs for each audio frame). We replicated this with each of the seven methods under comparison, fusing the classification results from the four representations late in the procedure for each method. These results are listed in Table VI. All methods were improved by using binaural audio, but either DCAR or DCAR(c) was still consistently superior to the five baselines in terms of $FScore$ and $MissRate$.

Finally, $Accuracy$ scores for each class are given in Table VII (averaged across the four folds). The results are separated into three groups: vehicle, indoor, and outdoor. Our methods obtain competitive performance for all three groups, and either DCAR or DCAR(c) obtains one of the top two results for most classes (with the exception of Home and City Center). This may be because the embedding $\mathbf{W}$ is learned for all classes rather than for each individual class.

VIII. CONCLUSION AND FUTURE WORK

In this article, we have presented a new audio representation, DCAR, and demonstrated its use in event classification and audio-scene classification. One distinguishing characteristic of the DCAR method is that it can capture the variability within each audio track. Another is that it achieves better discriminative ability by integrating label information and the graph of the components' nearest neighbors among the audio tracks, i.e., it can successfully characterize both global and local structure among audio tracks.

Representing audio using the proposed DCAR notably improves performance on event detection (using the YLI-MED dataset) and audio scene classification (using the DCASE challenge dataset), as compared with the existing popular representations (e.g., on YLI-MED, an average of more than 8% relative gain for ten-event classification and more than 10% gain for binary detection across all metrics).

The proposed representation benefits from leveraging global and local structure within audio data; however, videos are of course *multi*modal. Other data sources such as visual content, captions, and other metadata can provide valuable information for event detection; we therefore plan to extend the current model by incorporating such information.

Related work on audio analysis (e.g., Coviello *et al.,* 2011 [36]; Coviello *et al.*, 2012 [37]; Barchiesi *et al.,* 2015 [8]) has demonstrated that the temporal evolution of different events plays an important role in audio analysis; another possible direction for expanding DCAR is to take into consideration complex temporal information in modeling video events.

Last but not least, we might explore extracting the information-rich segments from each audio track rather than modeling the whole track.

APPENDIX A
PROOF: THE ROTATIONAL INVARIANCE OF F

Given the objective function $\mathbf{F}(\mathbf{W})$ in (12), repeated here as (20), and any rotation matrix $\mathbf{R} \in SO(r)$ (i.e., $\mathbf{R}\mathbf{R}^T =$

$\mathbf{R}^T\mathbf{R} = \mathbf{I}_r$), we can show that $\mathbf{F}(\mathbf{W}) = \mathbf{F}(\mathbf{WR})$

$$\mathbf{F}(\mathbf{W}) = \sum_{i,j} A_{ij}\Big(\lambda\|\mathbf{W}^T(\mu_i-\mu_j)\|_F^2 + \|\log(\mathbf{W}^T\Sigma_i\mathbf{W}) - \log(\mathbf{W}^T\Sigma_j\mathbf{W})\|_F^2\Big). \tag{20}$$

Proof. According to the definition of $\mathbf{F}(\mathbf{W})$ in (20), we can write $\mathbf{F}(\mathbf{WR})$ as

$$\begin{aligned}\mathbf{F}(\mathbf{WR}) = \sum_{i,j} A_{ij}\Big(&\lambda\|(\mathbf{WR})^T(\mu_i-\mu_j)\|_F^2 \\ &+ \|\log((\mathbf{WR})^T\Sigma_i\mathbf{WR}) \\ &- \log((\mathbf{WR})^T\Sigma_j\mathbf{WR})\|_F^2\Big).\end{aligned}$$

We set

$$\begin{aligned}\mathbf{F1}_{ij}(\mathbf{WR}) &= \|(\mathbf{WR})^T(\mu_i-\mu_j)\|_F^2 \\ &= Tr\Big[(\mu_i-\mu_j)^T(\mathbf{WR})(\mathbf{WR})^T(\mu_i-\mu_j)\Big] \\ &= Tr\Big[(\mu_i-\mu_j)^T\mathbf{WRR}^T\mathbf{W}^T(\mu_i-\mu_j)\Big] \\ &= Tr\Big[(\mu_i-\mu_j)^T\mathbf{WW}^T(\mu_i-\mu_j)\Big] \\ &= \|(\mathbf{W})^T(\mu_i-\mu_j)\|_F^2 \\ &= \mathbf{F1}_{ij}(\mathbf{W}).\end{aligned} \tag{21}$$

Since $0 \prec \mathbf{W}^T\Sigma_i\mathbf{W} \in \mathcal{S}ym_r^+$ is symmetric positive-definite, and the log-euclidean metric has the properties of Lie group bi-invariance and similarity invariance), i.e.,

$$\|\log(X) - \log(Y)\|_F^2 = \|\log(\mathbf{R}^TX\mathbf{R}) - \log(\mathbf{R}^TY\mathbf{R})\|_F^2$$

then

$$\begin{aligned}\mathbf{F2}_{ij}(\mathbf{WR}) &= \|\log((\mathbf{WR})^T\Sigma_i\mathbf{WR}) \\ &\quad - \log((\mathbf{WR})^T\Sigma_j\mathbf{WR})\|_F^2 \\ &= \|\log(\mathbf{R}^T(\mathbf{W}^T\Sigma_i\mathbf{W})\mathbf{R}) \\ &\quad - \log(\mathbf{R}^T(\mathbf{W}^T\Sigma_j\mathbf{W})\mathbf{R})\|_F^2 \\ &= \|\log((\mathbf{W}^T\Sigma_i\mathbf{W}) - \log(\mathbf{W}^T\Sigma_j\mathbf{W})\|_F^2 \\ &= \mathbf{F2}_{ij}(\mathbf{W}).\end{aligned} \tag{22}$$

Thus

$$\begin{aligned}\mathbf{F}(\mathbf{WR}) &= \sum_{i,j} A_{ij}\big(\lambda\mathbf{F1}_{ij}(\mathbf{WR}) + \mathbf{F2}_{ij}(\mathbf{WR})\big) \\ &= \sum_{i,j} A_{ij}\big(\lambda\mathbf{F1}_{ij}(\mathbf{W}) + \mathbf{F2}_{ij}(\mathbf{W})\big) = \mathbf{F}(\mathbf{W})\end{aligned} \tag{23}$$

as claimed. ■

Appendix B
Solving the Optimization Problem in (12)

Let $\nabla_{\mathbf{W}}$ and $\mathcal{D}_{\mathbf{W}}$ be the tangent vector and the gradient of $\mathbf{F}(\mathbf{W})$ at point $\mathbf{W}$ of the Grassmannian manifold. The gradient at the τ-th iteration can be obtained by subtracting the normal component at $\mathbf{W}^{(\tau)}$ from the transported vector:

$$\mathcal{D}_{\mathbf{W}}^{(\tau)} = \nabla_{\mathbf{W}}^{(\tau)} - \mathbf{W}^{(\tau)}(\mathbf{W}^{(\tau)})^T\nabla_{\mathbf{W}}^{(\tau)}. \tag{24}$$

The search direction $\mathcal{H}_{\mathbf{W}}$ in the $(\tau+1)$-th iteration can be computed by parallel transporting the previous search direction and combining it with the gradient direction at the current solution

$$\mathcal{H}_{\mathbf{W}}^{(\tau+1)} = \mathcal{D}_{\mathbf{W}}^{(\tau+1)} + \gamma^{(\tau+1)}\triangle\mathcal{H}_{\mathbf{W}}^{(\tau)}. \tag{25}$$

$\triangle\mathcal{H}_{\mathbf{W}}^{(\tau)}$ is the parallel translation of the vector $\mathcal{H}_{\mathbf{W}}^{(\tau)}$. According to Absil, Mahony, and Sepulcher 2008 [28], the geodesic going from point $\mathbf{W}$ in the direction $\mathcal{H}_{\mathbf{W}}^{(\tau)}$ can be represented by the geodesic equation

$$\mathbf{W}(t) = \begin{bmatrix}\mathbf{W}\boldsymbol{V} & \boldsymbol{U}\end{bmatrix}\begin{bmatrix}\cos\boldsymbol{\Lambda}t \\ \sin\boldsymbol{\Lambda}t\end{bmatrix}V^T. \tag{26}$$

Here, t is a scalar that determines how far $\boldsymbol{W}(t)$ can walk along the geodesic in the descent direction. $\mathbf{F}(\mathbf{W}(t))$ is not a convex function over t, but minimizing it is a one-dimensional optimization problem. Thus, we adopted the derivative-free line search (e.g., golden-section search) to determine the optimal $t^{(\tau)}$ in the τ-th iteration. According to [38], this line search method can find the local optimal point.

Then, we have

$$\triangle\mathcal{H}_{\mathbf{W}}^{(\tau)} = \big(-\mathbf{W}^{(\tau)}\boldsymbol{V}\sin\boldsymbol{\Lambda}t^{(\tau)} + \boldsymbol{U}\cos\boldsymbol{\Lambda}t^{(\tau)}\big)\boldsymbol{\Lambda}\boldsymbol{V}^T \tag{27}$$

where $\boldsymbol{U\Lambda V}^T$ is the compact singular value decomposition of $\mathcal{H}_{\mathbf{W}}^{(\tau)}$.

The step size $\gamma^{(\tau+1)}$ can be determined via the exact conjugacy condition

$$\gamma^{(\tau+1)} = \frac{\langle\mathcal{D}_{\mathbf{W}}^{(\tau+1)} - \triangle\mathcal{D}_{\mathbf{W}}^{(\tau)}, \mathcal{D}_{\mathbf{W}}^{(\tau+1)}\rangle}{\langle\mathcal{D}_{\mathbf{W}}^{(\tau)}, \mathcal{D}_{\mathbf{W}}^{(\tau)}\rangle} \tag{28}$$

where $\langle A, B\rangle = Tr(A^TB)$. Similar to $\triangle\mathcal{H}_{\mathbf{W}}^{(\tau)}$, $\triangle\mathcal{D}_{\mathbf{W}}^{(\tau)}$ can be calculated by

$$\begin{aligned}\triangle\mathcal{D}_{\mathbf{W}}^{(\tau)} = \mathcal{D}_{\mathbf{W}}^{(\tau)} - \big(&\mathbf{W}^{(\tau)}V\sin\Lambda t^{(\tau)} + U(\mathbf{I} - \cos\Lambda t^{(\tau)})\big) \\ &\times U^T\mathcal{D}_{\mathbf{W}}^{(\tau)}.\end{aligned} \tag{29}$$

Going back to the objective function in (11), by setting $\mathbf{F1}_{ij} = \|\mathbf{W}^T(\mu_i-\mu_j)\|_F^2$ and $\mathbf{F2}_{ij} = \|\log(\mathbf{W}^T\Sigma_i\mathbf{W}) - \log(\mathbf{W}^T\Sigma_j\mathbf{W})\|_F^2$, (11) can be rewritten as

$$\mathbf{F}(\mathbf{W}) = \sum_{i,j} A_{ij}\big(\lambda\mathbf{F1}_{ij} + \mathbf{F2}_{ij}\big) \tag{30}$$

then its tangent vector $\nabla_{\mathbf{W}}$ on the manifold can be computed in three steps

$$\nabla_{\mathbf{W}}\mathbf{F1}_{ij} = 2(\mu_i-\mu_j)(\mu_i-\mu_j)^T\mathbf{W} \tag{31}$$

Algorithm 1: Solving (12) via a Conjugate Gradient on a Grassmannian Manifold

```
Input: A set of labeled $d$-dimensional GMM components
       $\{g_i, \ell_i\}_{i=1}^N$ with means $\{\mu_i\}_{i=1}^N$ and covariance matrices
       $\{\Sigma\}_{i=1}^N$, reduced dimensionality $r$, and parameter $\lambda$.
 1: Construct an affinity matrix $A$ using (5)
 2: Initialize $\boldsymbol{W}^{(0)}$ such that $(W^{(0)})^T W^{(0)} = \mathbf{I}_r$ and $\tau = 0$
 3: Compute $\nabla_{\mathbf{W}}^{(0)}$ as in (33) and $\mathcal{D}_{\mathbf{W}}^{(0)}$ as in (24), and set
    $\mathcal{H}_{\mathbf{W}}^{(0)} = -\mathcal{D}_{\mathbf{W}}^{(0)}$
 4: for $\tau = 0, 1, 2, \cdots$ do
 5:     Perform $[\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}] = SVD(\mathcal{H}_{\mathbf{W}}^{(\tau)})$
 6:     Given $\boldsymbol{U}$, $\boldsymbol{\Lambda}$, $\boldsymbol{V}$ and (26), find the optimal $\mathbf{W}^{(\tau)}(t)$
        over $t$ with golden-section search. Set $t^{(\tau)} = t_{\min}$
        (where $t_{\min}$ is the value of $t$ at the optimal point of
        $\mathbf{W}^{(\tau)}$)
 7:     Compute $\triangle\mathcal{H}_{\mathbf{W}}^{(\tau)}$ and $\triangle\mathcal{D}_{\mathbf{W}}^{(\tau)}$ as in (27) and (29)
 8:     Set $\mathbf{W}^{(\tau+1)} = \mathbf{W}^{(\tau)}(t^{(\tau)})$
 9:     Compute $\nabla_{\mathbf{W}}^{(\tau+1)}$ as in (33) and $\mathcal{D}_{\mathbf{W}}^{(\tau+1)}$ as in (24)
10:     Find the step size $\gamma^{(\tau+1)}$ via (28)
11:     Find the new search direction $\mathcal{H}_{\mathbf{W}}^{(\tau+1)}$ via (25)
12: end for
Output: $\boldsymbol{W}^{(\tau+1)}$
```

$$\nabla_{\mathbf{W}}\mathbf{F2}_{ij} = 4\Big(\Sigma_i \mathbf{W}(\mathbf{W}^T\Sigma_i\mathbf{W})^{-1} - \Sigma_j\mathbf{W}(\mathbf{W}^T\Sigma_j\mathbf{W})^{-1}\Big) \times \Big(\log(\mathbf{W}^T\Sigma_i\mathbf{W}) - \log(\mathbf{W}^T\Sigma_j\mathbf{W})\Big) \tag{32}$$

$$\nabla_{\mathbf{W}} = \sum_{i,j} A_{ij}\big(\lambda_1\nabla_{\mathbf{W}}\mathbf{F1}_{ij} + \lambda_2\nabla_{\mathbf{W}}\mathbf{F2}_{ij}\big). \tag{33}$$

This conjugate gradient method for solving (12) is summarized in Algorithm 1.

Appendix C
Complexity Analysis of the Methods Compared

Storage complexity: Suppose we have n tracks belonging to L classes, with each track represented by a collection of d-dimensional MFCC feature vectors ($d = 60$). For i-vector, the vector dimensionality is d^* (where d^* is tuned to one of the values in in $\{200, 400, 600, 800, 1000\}$). It requires $O(nd^*)$ of space to store the vectors. The storage complexity for mv-vector is $O(nd')$, with $d' = 131$. For GMM, each component contains a mean vector and a covariance matrix, thus it requires $O(nPd^2 + nPd)$ of space, where P is the number of components extracted from each track. For HEM-GMM, the storage complexity is $O(LP_Ld^2 + LP_Ld)$, where P_L is the number of component extracted from each class.

Similar to GMM, the storage for DCAR is $O(nPr^2 + nPr)$, where r is the reduced dimensionality; it is less than d. DCAR(c) requires $O(LP_Lr^2 + LP_Lr)$ of space. The compactness of DCAR and DCAR(c) can be compared with that of GMM and HEM-GMM, respectively. For example, it can be seen from Fig. 3(a) and Table II that, on the EC5 subset, DCAR obtains an accuracy comparable to plain GMM at $r = 25$. In that case, the storage complexity of DCAR/DCAR(c) is about $\frac{1}{6}$ that of GMM/HEM-GMM.

Time complexity: In each iteration of Algorithm 1, we calculate the gradient of the pairwise distances for a total of N^2 distances ($N = nP$ for DCAR, and $N = LP_L$ for DCAR(c)). Then compact SVD and matrix inversion must be implemented, with a time complexity of $O(r^3)$. Therefore, the time complexity for learning $\boldsymbol{W}$ is $O(N^2r^3)$. In our experiments, only the k-nearest neighbors are considered for each component, which makes the affinity matrix A very sparse. The number of non-zero distance pairs is thus many fewer than N^2. Meanwhile, r is less than 60 (say, $r = 25$ for EC5 and HC4). An efficient method is to compute the gradients in parallel and then sum them.

Classification complexity: Due to computing the kernel for components, the complexity of the classification algorithm is $O(N^2r^2 + Nr^3)$ for the training phase and $O(P(d^2r + Nr^2))$ for the testing phase. In the future, we plan to employ a more computationally efficient classification method (rather than a kernel classifier).

Acknowledgment

The authors are grateful to the anonymous TMM and ACMMM reviewers who made many helpful suggestions for improving this paper. (Any findings and conclusions are those of the authors, and do not necessarily reflect the views of the funders.)

References

[1] L. Jing *et al.*, “A discriminative and compact audio representation for event detection,” in *Proc. ACM Multimedia Conf.*, 2016, pp. 57–61.

[2] R. Cai, L. Lu, and A. Hanjalic, “Co-clustering for auditory scene categorization,” *IEEE Trans. Multimedia*, vol. 10, no. 4, pp. 596–606, Jun. 2008.

[3] A. Casanovas, G. Monaci, P. Vandergheynst, and R. Gribonval, “Blind audiovisual source separation based on sparse redundant representations,” *IEEE Trans. Multimedia*, vol. 12, no. 5, pp. 358–371, Aug. 2010.

[4] X. Valero and F. Alias, “Gammatone cepstral coefficients: Biologically inspired features for non-speech audio classification,” *IEEE Trans. Multimedia*, vol. 14, no. 6, pp. 1684–1689, Dec. 2012.

[5] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. Plumbley, “Detection and classification of audio scenes and events,” *IEEE Trans. Multimedia*, vol. 17, no. 10, pp. 1733–1746, Oct. 2015.

[6] A. Owens, J. Wu, J. McDermott, W. Freeman, and A. Torralba, “Ambient sound provides supervision for visual learning,” in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 801–816.

[7] G. Evangelopoulos *et al.*, “Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention,” *IEEE Trans. Multimedia*, vol. 15, no. 7, pp. 1553–1568, Nov. 2013.

[8] D. Barchiesi, D. Giannoulis, D. Stowell, and M. Plumbley, “Acoustic scene classification: Classifying environments from the sounds they produce,” *Signal Process. Mag.*, vol. 32, no. 3, pp. 16–34, 2015.

[9] A. Eronen, J. Tuomi, A. Klapuri, and S. Fagerlund, “Audio-based context awareness—Acoustic modeling and perceptual evaluation,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Oct. 2003, pp. 529–532.

[10] M. Ravanelli, B. Elizalde, J. Bernd, and G. Friedland, “Insights into audio-based multimedia event classification with neural networks,” in *Proc. Workshop Community-Organized Multimodal Mining, Opportunities Novel Solutions*, 2015, pp. 19–23.

[11] Q. Jin, P. Schulman, S. Rabat, S. Burger, and D. Ding, “Event-based video retrieval using audio,” in *Proc. INTERSPEECH*, 2012, pp. 2085–2088.

[12] R. Mertens, H. Lei, L. Gottlieb, G. Friedland, and A. Divakaran, “Acoustic super models for large scale video event detection,” in *Proc. ACM Joint ACM Workshop Model. Representing Events*, 2011, pp. 19–24.

[13] Z. Huang, Y. Cheng, K. Li, V. Hautamaki, and C. Lee, "A blind segmentation approach to acoustic event detection asked on i-vector," in *Proc. INTERSPEECH*, 2013, pp. 2282–2286.
[14] B. Elizalde, H. Lei, and G. Friedland, "An i-vector representation of acoustic environment for audio-based video event detection on user generated content," in *Proc. IEEE Int. Symp. Multimedia*, Dec. 2013, pp. 114–117.
[15] G. Roma, W. Nogueira, and P. Herrera, "Recurrence quantification analysis features for auditory scene classification," IEEE AASP Challenge, Detection and Classification of Acoustic Scenes and Events, 2013.
[16] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 467–476, Feb. 2008.
[17] N. Dehak, P. Kenny, R. Dehak, P. Dumouchefl, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 19, no. 4, pp. 788–798, May 2011.
[18] B. Scholkopf and A. Smola, Eds., *Learning With Kernels*. Cambridge, MA, USA: MIT Press, 2002.
[19] L. Jing, C. Zhang, and M. Ng, "SNMFCA: Supervised NMF-based image classification and annotation," *IEEE Trans. Image Process.*, vol. 21, no. 11, pp. 4508–4521, Nov. 2012.
[20] M. Lan, C. Tan, J. Su, and Y. Lu, "Supervised and traditional term weighting methods for automatic text categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 4, pp. 721–735, Apr. 2009.
[21] S. Hershey *et al.*, "CNN architectures for large-scale audio classification," presented at Int. Conf. Acoust., Speech, Signal Process., 2017.
[22] W. Wang, R. Wang, Z. Huang, S. Shan, and X. Chen, "Discriminant analysis on Riemannian manifold of Gaussian distributions for face recognition with image sets," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2015, pp. 2048–2057.
[23] G. McLachlan, Ed., *Discriminant Analysis and Statistical Pattern Recognition*. Hoboken, NJ, USA: Wiley, 2004.
[24] H. Li, T. Jiang, and K. Zhang, "Efficient and robust feature extraction by maximum margin criterion," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 97–104.
[25] X. Pennec, P. Fillard, and N. Apache, "A Riemannian framework for tensor computing," *Int. J. Comput. Vis.*, vol. 66, no. 1, pp. 41–66, 2006.
[26] A. Cherian, S. Sra, A. Banerjee, and N. Papanikolopoulos, "Efficient similarity search for covariance matrices via the Jensen-Bregman LogDet divergence," in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 2399–2406.
[27] V. Arsigny, P. Fillard, X. Pennec, and N. Apache, "Geometric means in a novel vector space structure on symmetric positive definite matrices," *SIAM Matrix Anal.*, vol. 29, no. 1, pp. 328–347, 2007.
[28] P. Absil, R. Mahony, and R. Sepulcher, *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ, USA: Princeton Univ. Press, 2008.
[29] J. Bernd *et al.*, "The YLI-MED corpus: Characteristics, procedures, and plans (TR-15-001)," Int. Computer Sci. Inst., Berkeley, CA, USA, Tech. Rep. TR-15-001, 2015.
[30] P. Over *et al.*, "TRECVID 2011—An overview of the goals, tasks, data, evaluation mechanisms, and metrics," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. {TR}-hal-00763912, May 2012. [Online]. Available: http://www-nlpir.nist.gov/projects/tvpubs/tv11.papers/tv11overview.pdf
[31] B. Thomee *et al.*, "YFCC100M: The new data in multimedia research," *Commun. ACM*, vol. 59, no. 2, pp. 64–73, 2016.
[32] L. Zelnik-manor and P. Perona, "Self-tuning spectral clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 1601–1608.
[33] I. Jolliffe, *Principal Component Analysis*. Hoboken, NJ, USA: Wiley, 2005.
[34] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *Proc. EUSIPCO*, 2016, pp. 1128–1132.
[35] H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, "CP-JKU submissions for DCASE-2016: A hybrid approach using binaural i-vectors and deep convolutional neural networks," *Detection and Classification of Acoustic Scenes and Events*, Budapest, Hungary, 2016. [Online]. Available: http://www.cs.tut.fi/sgn/arg/dcase2016/task-results-acoustic-scene-classification
[36] E. Coviello, A. B. Chan, and G. Lanckriet, "Time series models for semantic music annotation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 5, pp. 1343–1359, Jul. 2011.
[37] E. Coviello, Y. Vaizman, A. B. Chan, and G. R. G. Lanckriet, "Multivariate autoregressive mixture models for music auto-tagging," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, 2012, pp. 547–552.
[38] R. Brent, Ed., *Algorithms for Minimization Without Derivatives*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1973.

Liping Jing received the Ph.D. degree in applied mathematics from the University of Hong Kong, Hong Kong, China, in 2007.

She is currently a Professor with the Beijing Key Laboratory of Traffic Data Analysis and Mining, School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China. She was previously a Research Associate with the Department of Mathematics, Hong Kong Baptist University, Hong Kong, China, a Research Fellow with the Department of Computer Science, University of Texas at Dallas, Dallas, TX, USA, from 2007 to 2008, and a Visiting Scholar with ICSI and the AMPLab in University of California at Berkeley, Berkeley, CA, USA, from 2015 to 2016. She has authored or coauthored more than 70 peer-reviewed research papers in various journals and conferences. Her research interests include high-dimensional subspace learning and machine learning.

Bo Liu, photograph and biography not available at the time of publication.

Jaeyoung Choi, photograph and biography not available at the time of publication.

Adam Janin, photograph and biography not available at the time of publication.

Julia Bernd, photograph and biography not available at the time of publication.

Michael W. Mahoney, photograph and biography not available at the time of publication.

Gerald Friedland, photograph and biography not available at the time of publication.