A Systematic Evaluation of the Bag-of-Frames Representation for Music Information Retrieval

Li Su, *Member, IEEE*, Chin-Chia Michael Yeh, Jen-Yu Liu, Ju-Chiang Wang, and Yi-Hsuan Yang, *Member, IEEE*

***Abstract*—There has been an increasing attention on learning feature representations from the complex, high-dimensional audio data applied in various music information retrieval (MIR) problems. Unsupervised feature learning techniques, such as sparse coding and deep belief networks have been utilized to represent music information as a term-document structure comprising of elementary audio codewords. Despite the widespread use of such bag-of-frames (BoF) model, few attempts have been made to systematically compare different component settings. Moreover, whether techniques developed in the text retrieval community are applicable to audio codewords is poorly understood. To further our understanding of the BoF model, we present in this paper a comprehensive evaluation that compares a large number of BoF variants on three different MIR tasks, by considering different ways of low-level feature representation, codebook construction, codeword assignment, segment-level and song-level feature pooling, tf-idf term weighting, power normalization, and dimension reduction. Our evaluations lead to the following findings: 1) modeling music information by two levels of abstraction improves the result for difficult tasks such as predominant instrument recognition, 2) tf-idf weighting and power normalization improve system performance in general, 3) topic modeling methods such as latent Dirichlet allocation does not work for audio codewords.**

***Index Terms*—Bag-of-frames model, music information retrieval, sparse coding, unsupervised feature learning.**

I. Introduction

OVER the recent years, sparse coding (SC) and deep belief networks (DBNs) algorithms have been utilized for constructing the codebook for music [1]–[17]. Inspired by the human's sensory system, SC aims at forming codes that are sparse in support (with most coefficients being zero) but are sufficient to reconstruct or to interpret the input signal [1]. The codebook of SC can be pre-defined using standard bases such as wavelet, Gabor, or Gammatone functions [1]–[4], but can also be learnt from a collection of music signals using algorithms such as matching pursuit and online dictionary learning (ODL) [12], [18].[1] In contrast, DBN is probabilistic, multilayer neural network that processes information by multiple levels of transformation and abstraction, as different areas of cortex in the mammal brain perform [19]. The idea of DBN is to form a hierarchical signal processing paradigm that mimics how people organize and perceive music information.

Given a codebook, any acoustic feature vector can be replaced by the occurrence of codewords in the corresponding music signal, leading to the so-called *bag-of-frames* (BoF) representation of music [20]. This technique assumes that a vocabulary consists of finite words and that documents are unordered sets of word occurrences. Audio events local in time (e.g., guitar solo or riffs) can be represented by different codewords in the BoF model, instead of being smeared out as in the case of taking mean or median pooling over the entire feature sequence [21], [22]. Moreover, as the feature representation is text-like, one can recast MIR as text IR and benefit from the lessons and techniques that have been learnt and developed for text [23].

Although the codebook-based approaches have been shown powerful alternatives to hand-crafted feature design for music information retrieval (MIR) [17], little work has been done to systematically compare the performance of different methods on multiple MIR tasks. Instead of "reinventing the wheels," we report a comprehensive set of experiments that tried to evaluate the performance of existing BoF-based methods as accurately as possible. Moreover, many existing work adopted a "transductive learning" paradigm and assumed that the test set of the target task is available during feature learning (e.g., [14]), which is not always the case in practice. In view of this issue, we learn the codebook from an external data set (i.e., 6,700 tracks from USPOP [24]) that is disjoint from the data sets of the considered tasks, which include music genre classification [25], predominant instrument recognition [26], semantic annotation (i.e., auto-tagging) and retrieval [27].

In particular, this work has an explicit focus on studying the analogy between text words and audio codewords, an issue that has been poorly addressed before. Instead of simply counting the frequency of occurrence of the codewords, we study the influence of different keyword weighting schemes, using 25 different combinations of term frequency-inverse document frequency (tf-idf) measures [23]. These measures enhance the significance of terms that have high weight and occur sparsely in the whole corpora. Although the idea of tf-idf term weighting

Manuscript received November 19, 2012; revised June 03, 2013; accepted August 22, 2013. Date of publication March 11, 2014; date of current version July 15, 2014. This work was supported in part by the National Science Council of Taiwan under Grants NSC 101-2221-E- 001-017, NSC 102-2221-E-001-004-MY3 and in part by the Academia Sinica Career Development Award. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Tao Li.

L. Su, C.-C. M. Yeh, J.-Y. Liu, and Y.-H. Yang are with the Research Center for Information Technology Innovation, Academia Sinica, Taipei 11564, Taiwan (e-mail: lisu@citi.sinica.edu.tw; mcyeh@citi.sinica.edu.tw; ciaua@citi.sinica.edu.tw; yang@citi.sinica.edu.tw).

J.-C. Wang is with the Institute of Information Science, Academia Sinica, Taipei 11564, Taiwan (e-mail: asriver@iis.sinica.edu.tw)

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.2014.2311016

[1]Whether to favor a generic codebook or a codebook adapted to (i.e., learnt from) the data has been found task-dependent [2], [8].

TABLE I
COMPARISON OF BAG-OF-FRAMES BASED APPROACHES FOR MIR PROBLEMS; TR INDICATES TRANSDUCTIVE LEARNING

Work	Method(s)	Num. layer	Feature(s)	Task(s)	Corpus for feature learning	Key finding(s)
Riley (2008) [28]	VQ ($k$means / HAC / GMM)	1	Chromagram	Cover song	TR	The 3 algorithms performs similarly
Seyerlehner (2008) [33]	VQ ($k$means)	1	Spectrogram	Similarity	TR	Comparable to state-of-the-art
Fu (2011) [30]	VQ ($k$means)	1	MFCC	Genre	TR	81.7% for GTZAN
Wulfing (2012) [34]	VQ ($k$means)	1	Spectrogram	Genre	TR	85.3% for GTZAN
Mcfee (2012) [35]	VQ ($k$means)	1	MFCC	Similarity	Training split	Improves state-of-the-art
Weiss (2010) [36]	NMF	1	Chromagram	Segmentation	TR	Comparable to state-of-the-art
Wang (2011) [37]	GMM / VQ	1	MIR features	Similarity	TR	GMM slightly better than VQ
Lee (2009) [5]	DBN	2	Spectrogram	Genre	ISMIR2004	73.1% for GTZAN
Hamel (2010) [6]	DBN	3	Spectrogram	Genre	Training split	84.3% for GTZAN
Nam (2011) [7]	DBN	2	Spectrogram	Multi-pitch	TR	Comparable to state-of-the-art
Scluter (2011) [38]	DBN / VQ ($k$means)	4/1	Mel-spectrum +PCA	Similarity	TR	1) Comparable to state-of-the-art 2) DBN slightly better than VQ
Dieleman (2011) [9]	DBN	4	EchoNest timbre+chroma	Genre / artist / key	MSD	Outperform some baseline methods
Schmidt (2012) [11]	DBN	3	Spectrogram	Emotion	USPOP	Slightly improve state-of-the-art that uses hand-crafted features
Nam (2012) [14]	DBN / SC / VQ ($k$means)	1	PMSC	Auto-tagging	TR	1) Comparable to state-of-the-art 2) The 3 algorithms perform similarly
Henaff (2011) [10]	SC (PSD)	1	CQT	Genre	Training split	80.0% for GTZAN
Scholler (2011) [8]	SC (exemplar/MP)	1	Gammatone	Drum	ENST	Exemplar better than MP
Lee (2012) [16]	SC (exemplar)	1	Spectrogram	Multi-pitch	RWC	Comparable to state-of-the-art
Yeh (2012) [13]	SC (ODL) / SC (exemplar) / VQ ($k$means)	1	Spectrogram / Mel-spectrum / MFCC / CQT	Genre	TR	1) 84.7% for GTZAN 2) SC significantly better than VQ 3) SC+Spectrogram performs the best
Yeh (2013) [31]	SC (ODL)	2	Spectrogram	Genre	TR / USPOP	1) TR performs slightly better 2) Two-layer SC is sometimes better
This work	SC (ODL) / SC (exemplar) / VQ ($k$means)	2	Spectrogram / Mel-spectrum / PMSC	Genre / instrument / auto-tagging	USPOP	1) SC+ODL performs the best 2) Comparable to state-of-the-art 3) No need to use TR

has been applied to BoF models before [28]–[30], only the common variants of tf-idf has been utilized so far. Which weighting scheme is most suited to describe audio codewords deserves detailed assessments.

As audio codewords are usually computed from a single or a limited number of consecutive frames (e.g., less than 0.5 second) [4], [5], [14], it is reasonable to argue that such codewords correspond only to temporal audio elements or particles, instead of basic semantic units that can be quantified as words. Therefore, we are interested in extending the SC paradigm by cascading two layers of abstraction [31], one at the *alphabet level* (e.g., one frame), the other at the *word level* (e.g., 5-second segment). In this way, it is possible to encode multi-layer information as the DBN paradigm.

Moreover, probabilistic topic models such as latent Dirichlet allocation (LDA) [32] and its variants have also been widely used in text IR to uncover the hidden thematic structure in large collection of documents. Such techniques also reveal lexical relationship such as polysemy and synonymy.[2] We are therefore also interested in exploring the use of such probabilistic topic modeling to audio codewords.

The paper is organized as follows. Section II reviews related work. Section III starts with an overview of the proposed system and then goes into details of each system components. Section IV is dedicated to text-like techniques such as tf-idf weighting, latent Dirichlet allocation, and power normalization. Section V describes the data sets, low-level features, and the setup of the experimental parameters. Experiment results are reported in Section VI, followed by discussions in Section VII. Finally, Section VIII concludes the paper.

[2]Polysemy refers to a word that has multiple senses and multiple types of usage in different contexts, whereas and synonymy refers to different words that share a similar meaning) between words.

II. RELATED WORK

BoF models have been widely used in MIR, as Table I shows.[3] A classic approach to form an audio analogy of a word is by clustering a collection of frame-level feature vectors and using the cluster centers to form the codebook. This *vector quantization* (VQ) technique has been widely used [39]. For example, Riley *et al.* [28] compared three clustering algorithms for VQ and found that $k$means achieves competitive result with relatively less computational cost. VQ has been applied to genre classification and achieved 81.7% and 85.3% in ten-class classification for the GTZAN data set [30], [34] (both used transduction learning). Seyerlehner *et al.* [33] proposed a multi-level approach to accelerate VQ, whereas McFee *et al.* [35] used a soft variant of $k$means to reduce quantification errors. Other single-layer codebook learning methods such as nonnegative matrix factorization (NMF) [36] and GMM [37] and have also been proposed, among others.

To the best of our knowledge, Lee *et al.* [5] reported the first study that applies DBN to MIR problems. Using the features learnt by DBN outperforms standard acoustic features such as

[3]In the 'corpus' column, 'training split' indicates the training set of the target MIR task (e.g., genre classification) is used for dictionary learning. Moreover, we use 'TR' (i.e., transductive learning) to indicate the case where the test set of the target MIR task is also used for dictionary learning.

TABLE II
SUMMARY OF AUDIO-WORD AND AUDIO-ALPHABET NOTATIONS

Scale	Formulation	Notation	Dimension
Frame level (alphabet; e.g., 46 ms) (elementary unit)	$\hat{\alpha}_{\mathbf{s},\mathbf{t}} = \arg\min_{\alpha_{s,t}} \|\mathbf{x}_{s,t} - \mathbf{D}_1\alpha_{s,t}\|_2^2 + \lambda f(\alpha_{s,t})$	$\mathbf{D}_1$: first-layer codebook of audio-alphabets $\mathbf{x}_{s,t}$: frame-level acoustic feature $\alpha_{s,t}$: frame-level encoding over audio-alphabets	$\mathbf{D}_1 \in \mathbb{R}^{m\times k_1}$ $\mathbf{x}_{s,t} \in \mathbb{R}^m$ $\alpha_{s,t} \in \mathbb{R}^{k_1}$
Segment level (word; e.g., 2.5 sec) (combination of alphabets)	$\hat{\beta}_{\mathbf{s}} = \arg\min_{\beta_s} \|\sum_t \alpha_{s,t} - \mathbf{D}_2\beta_s\|_2^2 + \lambda f(\beta_s)$	$\mathbf{D}_2$: second-layer codebook of audio-words level $\overline{\alpha}_s = \sum_t \alpha_{s,t}$: pooled frame-level encoding β_s: segment-level encoding over audio-words	$\mathbf{D}_2 \in \mathbb{R}^{k_1\times k_2}$ $\overline{\alpha}_s \in \mathbb{R}^{k_1}$ $\beta_s \in \mathbb{R}^{k_2}$
Clip level (document; e.g., 30 sec)	$\gamma_1 = \sum_s \overline{\alpha}_s = \sum_{s,t} \alpha_{s,t}$ $\gamma_2 = \sum_s \beta_s$	γ_1: bag-of-audio alphabet (BoAA) γ_2: bag-of-audio word (BoAW)	$\gamma_1 \in \mathbb{R}^{k_1}$ $\gamma_2 \in \mathbb{R}^{k_2}$

spectrogram and MFCC for both genre classification and singer identification. Using a three-layer DBN, Hamel and Eck [6] obtained 84.3% accuracy in genre classification for the GTZAN data set. Many variants of DBN, such as convolutional DBN [9], conditional DBN [15], and convolutional neural networks (CNN) [40], [41], have also been utilized for MIR. Readers are referred to [17] for a recent overview.

Despite the effectiveness of DBN, Hamel *et al.* [6] noted that DBN requires large number of hyperparameters to be tuned and possible longer training times. In contrast, efficient approximated SC method such as predictive sparse decomposition (PSD) has been developed [10]. Moreover, unlike DBNs, it is easier to use pre-defined codebooks for SC, which has been shown advantageous over learnt codebooks for tasks such as multipitch detection [16] and drum classification [8]. For other MIR tasks such as similarity estimation and auto-tagging, it has been shown that the performance of DBN and SC is similar [38] and [14].[4] Therefore, we focus on the SC-based approach in this paper.

People are interested in sparse representations or sparse models, because they lead to a clear interpretation [12]. As first demonstrated by Smith *et al.* [1], audio codewords learnt by using the matching pursuit (MP) algorithm for sparse decomposition show striking similarities to time-domain cochlear filter estimates. Furthermore, as shown by Henaff *et al.* [10], codewords learnt from Constant-Q representations (CQT) using SC correspond well to the specific chords or pitch intervals such as minor thirds, perfect fifths, sevenths, major triads, etc. Therefore, in addition to the discriminative power, SC leads to codewords that are better in interpretability.

Several SC algorithms have been utilized in the literature for representing music information. For example, Henaff *et al.* obtained 80% in genre classification for the GTZAN data set (in a non-transductive, or *inductive*, scenario) by using PSD [10]. Scholler and Purwins [8] found that using a gammatone dictionary as exemplar codewords leads to better accuracy in drum sound classification than the codewords learnt by using matching pursuit [8]. Yeh *et al.* [13] found that using log-power spectrogram for low-level feature representation and ODL for feature learning leads to the best performance (84.7% for GTZAN, in a transductive learning scenario), whereas learning features from MFCC vectors using VQ does not perform well (70.1%). However, using $k$means for codebook learning but SC algorithms such as LARS-lasso [43] for encoding, the performance improves to 81.4%.[5] This result implies that the way the codewords are assigned may be more important than the way the codebook is generated, which is in line with the observations made in [42].

[4]It has been argued that the main value of the codebook might be to provide a highly overcomplete basis to which data are projected, and that the choice of algorithms for feature learning does not always matter [42]. Sometimes, even using random patches sampled from data as codewords can work well, as long as the codebook size is big enough [34].

TABLE III
VARIANTS OF TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY

Abbr.	Formulation	Abbr.	Formulation
tf_b	$f_{d,t}$	idf_b	$\ln\ \|\mathcal{D}\|/F_t$
tf_{l1}	$\ln(1+f_{d,t})$	idf_p	$\ln\ (\|\mathcal{D}\| - F_t)/F_t$
tf_{l2}	$1+\log_2 f_{d,t}$	idf_{e1}	$(\max_{t'\in\mathcal{T}} n_{t'}) - n_t$
tf_{o1}	$\frac{f_{d,t}}{f_{d,t}+\kappa\|d\|/\|\Delta d\|}$	idf_{e2}	$1 - \frac{n_t}{\ln\|\mathcal{D}\|}$
tf_{o2}	$\frac{(\kappa+1)f_{d,t}}{f_{d,t}+\kappa\left((1-b)+b\cdot\frac{\|d\|}{\|\Delta d\|}\right)}$	idf_o	$\ln\left(\frac{\|\mathcal{D}\|-F_t+0.5}{F_t+0.5}\right)$

$f_{d,t}$ — the number of occurrences of term t in document d
F_t — the number of documents in $\mathcal{D}$ that contain t
$|d|$ — cardinality (length) of d; $|\Delta d|$ — average document length in $\mathcal{D}$
n_t — entropy of term t in $\mathcal{D}$; $n_t = -\sum_d (f_{d,t}/F_t)\ln(f_{d,t}/F_t)$
$\mathcal{T}$ — the universe of terms; $\mathcal{D}$ — the universe of documents

Two-layer SC has been proposed in our prior work [31]. Our preliminary evaluation on GTZAN showed that the two-layer structure leads to the best accuracy (85.7%) for the transductive scenario, but the single-layer structure performs better (83.3%) for the inductive scenario. In addition, when BoF features are used, one can obtain result comparable to histogram intersection kernel (HIK) SVM [44] using just linear SVM, which reduces the processing time by more than 10 fold [31]. It has also been found feasible to use a fixed-length window for the second-layer music segmentation. The present paper adopts the findings of [31] and investigates many more other design parameters (see Table IV) for not only genre classification but more MIR tasks.

III. HIERARCHICAL FEATURE LEARNING FRAMEWORK

A. System Overview

Fig. 1 shows the flow diagram of the system we implemented for BoF-based MIR. The system makes use of a music collection, which we coin as the "training corpus," to build the audio codebook. This involves multiple operations such as encoding, pooling, weighting, and power normalization, which will be described later. Given the training clips of a target MIR task, the codebooks are used to obtain the BoF representation of the

[5]Different combinations of codebook generator and codeword encoder lead to different encoding systems. For the conventional VQ it uses a winner-takeall, nearest-neighbor approach for encoding, rather than sparse coding.

TABLE IV
THE DESIGN CHOICES AND PARAMETERS EVALUATED IN THIS PAPER, TOGETHER WITH THEIR EMPIRICAL OPTIMAL SETTING

Design choices/parameters	Options	Optimal choice (by result)
Low-level acoustic feature	STFT, MEL(128), PMSC(128), PMSC(80)	STFT
First-layer codebook learning method Second-layer codebook learning method	Exemplar, *k*means, ODL Exemplar, *k*means, ODL	ODL ODL
First-layer encoding method Second-layer encoding method	1-NN, 10-NN, tri-NN, L2, L1 1-NN, 10-NN, tri-NN, L2, L1	L1 L1
Segment-level pooling method (frames $\rightarrow$ segment) Clip-level pooling method (segments $\rightarrow$ clip)	Max, sum Max, sum	Max Sum
Type of term frequency functions Type of inverse document frequency functions	tf_b, tf_{l1}, tf_{l2}, tf_{o1}, tf_{o2} idf_b, idf_p, idf_{e1}, idf_{e2}, idf_o	tf_{l1} idf_{e2}
Power normalization	Square root, none	Square root
Dimension reduction	LDA, PCA	PCA

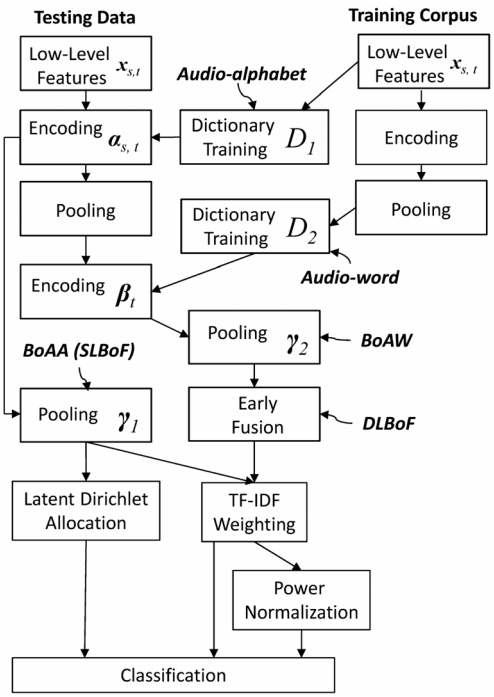

Fig. 1. The dual-layer feature learning framework with BoF model.

training clips, after which machine learning algorithms such as SVM are employed to train a classifier that takes the BoF features of test clips and performs classification. We adopted the *inductive learning* scenario and used a training corpus that has no overlap with both the training and test sets of the target MIR tasks.

Specifically, the system begins with extracting frame-level features such as spectrogram from the input music clip, which is divided into T segments of frames. It assumes that every segment contains the same number of S frames and uses $\mathbf{x}_{s,t} \in \mathbb{R}^m$ to denote the feature vector for the s-th frame of the t-th segment of the clip, where m is the size of the feature vector. After a sequence of operations, the information contained in the *frame-level* features is transformed and summarized as a *clip-level* histogram denoted as $\gamma \in \mathbb{R}^k$, where k is the number of histogram bins, or the codebook size.

For the case of the conventional, single-layer scheme, we constructed a codebook $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_k]$ directly from the frame-level feature vectors of the training corpus. As each codeword $\mathbf{d}_i \in \mathbb{R}^m$ lies in the same *signal space* as the acoustic feature vectors, one can use encoding methods such as sparse coding to express a feature vector $\mathbf{x}_{s,t}$ as a linear combination $\alpha_{s,t} \in \mathbb{R}^k$ of the codewords, a procedure that effectively transforms a datum in the signal space to the *indexical space* spanned by the audio codewords.[6] The clip-level BoF representation of a music signal can then be obtained by summarizing the sequences of coefficients $\alpha_{s,t}$ by temporal pooling functions such as summing over time, leading to $\gamma_1 = \sum_{s=1}^{S} \sum_{t=1}^{T} \alpha_{s,t}$.

As for the two-layer structure, we performed encoding and pooling in two levels: frame level and the intermediate segment level. The first-layer codebook $\mathbf{D}_1 \in \mathbb{R}^{m \times k_1}$ is learnt from the acoustic feature vectors, as done in the single-layer scheme. However, this time $\alpha_{s,t}$ is pooled over each segment instead of over the entire clip, giving rise to $\overline{\alpha}_s = \sum_{t=1}^{T} \alpha_{s,t}$ for the s-th segment of the clip, in the case of mean pooling (other pooling functions can also be used). The second-layer codebook $\mathbf{D}_2 \in \mathbb{R}^{k_1 \times k_2}$ is learnt from such intermediate encodings pooled at the segment level. As each codeword in $\mathbf{D}_2$ is also in the k_1-dimensional indexical space, one can express $\overline{\alpha}_s$ as a linear combination $\beta_s \in \mathbb{R}^{k_2}$, which essentially lies in another indexical space. Finally, another pooling over β_s leads to the clip-level histogram $\gamma_2 = \sum_{s=1}^{S} \beta_s$.

Based the analogy between alphabets and words, we refer to the (frame-level) codewords in $\mathbf{D}_1$ as the *audio-alphabets* and the (segment-level) codewords in $\mathbf{D}_2$ as the *audio-words*, respectively. Pooling audio-alphabet and audio-words features over the entire clip leads to the clip-level *bag-of-audio alphabet* (BoAA) γ_1 and the clip-level *bag-of-audio word* (BoAW) γ_2, respectively. Please see Fig. 1 and Table II for a summary, in which $f(\cdot)$ is a regularization term that will be described later.

BoAA and BoAW can be concatenated to form a more comprehensive BoF features. We refer to BoAA as *single-layerBoF* (SLBoF) and the concatenation of BoAA and BoAW as *double-layerBoF* (DLBoF) in the following discussion.

B. Codebook Construction

The codebook can be built from the acoustic feature vectors $\mathbf{x}_{s,t}$ or the pooled frame-level encoding $\overline{\alpha}_s$, depending on the

[6]The term signal space here refers to the span of dictionary atoms constructed by frequency spectrum basis whereas indexical space refers to feature vectors that are actually combination coefficients rather than features directly extracted from the signal.

abstraction level. For simplicity, we refer to both $\mathbf{x}_{s,t}$ and $\overline{\alpha}_s$ as "features" in this subsection. A total number of three codebook construction methods are considered.

1) Exemplar-Based Method: simply chooses a subset of feature vectors from the training corpus at random to form the codebook, a strategy that has been shown feasible for tasks such as denoising, face recognition [45], speech recognition [46], and multipitch detection of polyphonic music [16]. As there is no learning process, the computational cost is low, and it is easy to update the codebook. In addition, semantic interpretation is relatively straightforward for there is a one-to-one mapping between features and codewords.

2) The $k$means Algorithm: generates a codebook by selecting the k cluster centroids among a collection of feature vectors as codewords, assuming that the distribution of the features follows a Gaussian distribution around each codeword. For efficiency, the mini-batch variant [47] is adopted.

3) Online Dictionary Learning (ODL): is a first-order stochastic gradient descent algorithm proposed by Mairal *et al.* [18] to solve the following optimization problem,

$$\underset{\mathbf{D}\in\mathcal{C},\mathbf{Q}\in\mathbb{R}^{k\times n}}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\|\mathbf{p}_i-\mathbf{D}\mathbf{q}_i\|_2^2+\lambda f(\mathbf{q}_i)\right), \quad (1)$$

where $\mathbf{p}_i \in \mathbb{R}^r$ denotes the (observed) i-th signal among a set of n signals, $\mathbf{Q}=[\mathbf{q}_1,\ldots,\mathbf{q}_n]$ is the (unknown) encoding coefficients, $\mathbf{q}_i \in \mathbb{R}^k$, $\mathcal{C}$ is a set of (unknown) convex matrices $\mathbf{D}\in\mathbb{R}^{r\times k}$ satisfying $\mathbf{d}_j^T\mathbf{d}_j \le 1$, a constraint that is imposed to limit the energy of the codewords, $f(\mathbf{q})=\|\mathbf{q}\|_1$ is a regularization term that enforces the sparsity (here the l_1 norm) of the coefficients, and λ is a pre-set parameter for the trade-off between the sparsity of $\mathbf{q}$ and the representation accuracy.[7] A natural solution to this joint optimization problem is to solve for the two variables $\mathbf{D}$ and $\mathbf{q}$ in an alternating fashion [18]: minimize one while keeping the other fixed. The optimization of $\mathbf{D}$ uses block coordinate descent with warm restarts, which aggregates the past information computed during the previous steps of the algorithm. The optimization of $\mathbf{q}$ involves a typical sparse decomposition problem that is described in Section III-C1. Several optimization steps are made until convergence. The readers are referred to [18] for more details. Because of its low memory consumption and computational cost, ODL is more scalable than standard second-order matrix factorization algorithms such as K-SVD [48].

C. Encoding Methods

Given the codebook, any input signal can be represented by a linear combination of the codewords. The vector of combination coefficients can be either sparse or dense, depending on how the encoding algorithm manipulates the loss function and the sparsity constraint, as described below.

1) l_1-Regularized Sparse Coding (L1): The sparse coding problem can be described as

$$\hat{\mathbf{q}}=\arg\min_{\mathbf{q}}\|\mathbf{p}-\mathbf{D}\mathbf{q}\|_2^2+\lambda\|\mathbf{q}\|_1. \quad (2)$$

[7]According to the classical normalization factor [18], λ is set to $r^{-1/2}$ in this work, where r is the feature dimension of $\mathbf{p}_i$.

This problem is usually referred to as basis pursuit [49] or lasso [50] in the literature. It can be solved efficiently by off-the-shelf programs such as LARS-lasso [43].

2) l_2-Regularized Dense Coding (L2): or the Tikhonov regularization problem, concerns with

$$\hat{\mathbf{q}}=\arg\min_{\mathbf{q}}\|\mathbf{p}-\mathbf{D}\mathbf{q}\|_2^2+\lambda\|\mathbf{q}\|_2^2. \quad (3)$$

This is the classic l_2-regularized least-square error problem, whose solution corresponds to the minimum energy one. The use of l_p-norms (or $f(\mathbf{q})=\sum_r|\mathbf{q}|^p$, where $|\cdot|$ takes the absolute values) with $p>1$ leads to *dense* solutions [12].

3) Top-τ Vector Quantization (τ-NN): Conventional VQ performs encoding by searching for the unit vector that is nearest to the input feature vector in terms of Euclidean distance. In other words, this procedure can be regarded as a special case of a l_0-regularized least-square error problem, with the constraint $\|\mathbf{q}\|_0=1$ (number of non-zero terms being only one). One obvious extension is to find the τ-nearest unit vectors and take *multiple* quantizers for encoding [35].[8] This can be expressed as

$$\hat{\mathbf{q}}=\arg\min_{\mathbf{q}}\|\mathbf{p}-\mathbf{D}\mathbf{q}\|_2^2, \text{ subject to } \|\mathbf{q}\|_0=\tau \quad (4)$$

and all non-zero elements in $\mathbf{q}$ are assigned equal weighting in this work. It reduces to conventional VQ when τ is set to one. To avoid possible confusion with the codebook learning method $k$means or VQ, we denote this method as τ-NN.

4) Triangle $k$means (tri-NN): is another soft version of conventional VQ proposed in [51]. It compares the distance between the input feature vector with all the codewords and keeps those codewords whose distance to the input vector is less than the mean distance, or

$$q_j=\max\{0,\mu(\mathbf{z})-\mathbf{z}_j\}, \quad (5)$$

where $z_j=\|\mathbf{p}-\mathbf{d}_j\|_2$, $\mu(\mathbf{z})$ is the mean of $\mathbf{z}\in\mathbb{R}^k$, and z_j and q_j are the j-th element of $\mathbf{z}$ and $\mathbf{q}$, respectively.

D. Segmentation

It has been found that partitioning a music signal into short segments, each spanning a few frames, and then generating features based on the segmentation usually improves the accuracy of music categorization [25]. These segments, coined as "texture windows" [25], correspond to the minimum amount of time that is necessary to identify a particular music texture. One can segment a song by boundary detection algorithms [52] and dividing the song into nearly homogeneous parts with variable lengths. However, we followed [31] and segment a song with a window of constant length to capture the information of both homogeneous and inhomogeneous parts.

E. Temporal Pooling

Pooling is one of the most simple ways of aggregating time-varying information [21]. In the two-layer architecture, pooling

[8]This extension is intended to reduce the errors of VQ, which happens if a feature vector has multiple and approximately equidistant quantizers (codewords), leading to an unstable encoding [36]. This quantization error is more likely to happen as the size of the dictionary increases.

takes place in two stages: from multiple frames to a segment, and from multiple segments to a clip. In addition to sum pooling, we also experiment with *max pooling*, which extracts the maximum values of each component over all feature vectors in the time duration under consideration.

The effect of pooling methods on system performance has been studied in computer vision and MIR before. While some studies found it advantageous to use max pooling in the case of sparse features [14], [53], others showed that in practice both max pooling and sum pooling can be sub-optimal and that better result can be achieved by something in between [54]. In [55], max pooling is found suitable for sparse features through theoretical analysis, but its efficiency depends on other parameters such as pooling cardinality and codebook sizes.

IV. Textual Processing of Features

A. Term Frequency-Inverse Document Frequency (TF-IDF)

The tf-idf weighting is arguably one of the most important and widely used concepts in text IR [23], [56]. The tf term embodies the intuition that the more often a term occurs in a document, the more it is representative of its content. In contrast, the idf term puts emphasis on terms that occur in only a few number of documents across the corpus and thereby suppresses irrelevant terms such as function words.

Many variants of tf-idf weighting schemes have been proposed, and each of them has specific properties [57]. As Table III shows, we selected and compared the performance of 5 tf and 5 idf functions for BoF-based MIR, amounting to 25 possible combinations. Among them, tf_b, idf_b represent the basic formulations commonly applied in a variety of problems, tf_{l1}, tf_{l2} are two logarithmic variants, tf_{o1}, tf_{o2}, and idf_o are the Okapi formulations [58], idf_p is a probabilistic variant, and idf_{e1}, idf_{e2} are two information-theoretic entropy measures. As a comprehensive presentation of the tf-idf terms is beyond the scope of this paper, readers are referred to [56] and [59] and the references therein for more information.[9]

B. Latent Dirichlet Allocation (LDA)

LDA is a generative model that is originally proposed for topic modeling of texts [32]. In LDA, each document is assumed to be generated by a series of probabilistic processes which are controlled by several parameters. The model parameters can be learnt by the expectation-maximization (EM) algorithm derived from variational inference. Because audio codewords are text-like features, it might be feasible to apply LDA to the audio codewords and use the estimate of the topic distribution as the final feature representation. In this context, a term refers to an audio word and a document refers to a piece of music in LDA modeling.

LDA and tf-idf weighting cannot be applied together, because the input of LDA has to be in the tf_s form (i.e., a co-occurrence matrix that simply counts the number of occurrences of a term in a document). As Fig. 1 shows, LDA and tf-idf weighting were evaluated separately.

C. Power Normalization

Power transformation techniques are used in order to suppress anomalies of data, such as non-additivity, non-normality, and heteroscedasticity. When the pooled features are far from Gaussian distribution with non-additive error structure, this operation helps normalize the data. Given an input feature vector $\mathbf{q} \in \mathbb{R}^k$, the power normalization can be calculated with $\mathrm{sign}(\mathbf{q})|\mathbf{q}|^a$, where $\mathrm{sign}(\cdot)$ refers to signum function and $a \in [0, 1]$ is a parameter that is set to 0.5 in this paper according to Jégou *et al.* [60]. In other words, the BoF features are square rooted before being applied to a MIR task.

V. Experimental Setup

A. Problems

1) Genre Classification: Musical genres are the main top-level descriptors used by music dealers and librarians to organize music. Genre classification has also been extensively studied [52]. We used the frequently used GTZAN data set in this work.[10] It is composed of 1,000 30-second clips covering 10 genres (blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, rock), with 100 clips per genre. Although the correctness of the data set has been argued in recent papers (e.g., [61]), it provides a benchmark on which music classification algorithms can be compared.

2) Predominant Instrument Recognition: Instrument recognition is also related to music timbre. We made use of the data set collected by the Music Technology Group (MTG) of Universitat Pompeu Fabra,[11] which consists of approximately 2,500 excerpts of Western music labeled into 11 classes of pitched instruments (cello, clarinet, flute, acoustic guitar, electric guitar, Hammond organ, piano, saxophone, trumpet, violin and singing voice) and two classes of drums (drums and no-drums) [26]. The class labels are applied to the predominant instrument over a 3-second snippet of polyphony music.

3) Auto-Tagging: Recent research has focused on building detectors for tagging music with semantic labels such as genre, style, mood, and acoustic qualities [52]. For this evaluation, we used CAL500 [27],[12] a collection of 502 Western popular songs manually annotated with a lexicon of 174 pre-defined tags. The length of a clip ranges from 3 seconds to more than 22 minutes. According to the common protocol in the literature [14], we focused on a subset of 97 tags and evaluate the performance for both semantic annotation (multi-label classification) and retrieval.

[9]We provide some information about these functions here: the logarithmic operation in tf_{l1} and tf_{l2} is used to scale the effect of unfavorably high term frequency in one document; Okapi BM25 is a famous function for ranking documents according to their relevance to a given search query [58]. The parameters $\kappa \geq 0$ and $b \in [0, 1]$, whose optimal values have been found 1.2 and 0.75 for text IR, control the saturation rate of terms and document length normalization, respectively; the idf_p term, which is derived from a probabilistic model, measures the ratio between the number of documents that do not and do contain a term; both idf_{e1}, idf_{e2} reduce the importance of noisy (high entropy) terms.

[10]http://marsyas.info/download/data_sets

[11]http://www.dtic.upf.edu/ ffuhrmann/PhD/data/

[12]http://mkl.ucsd.edu/dataset/cal500

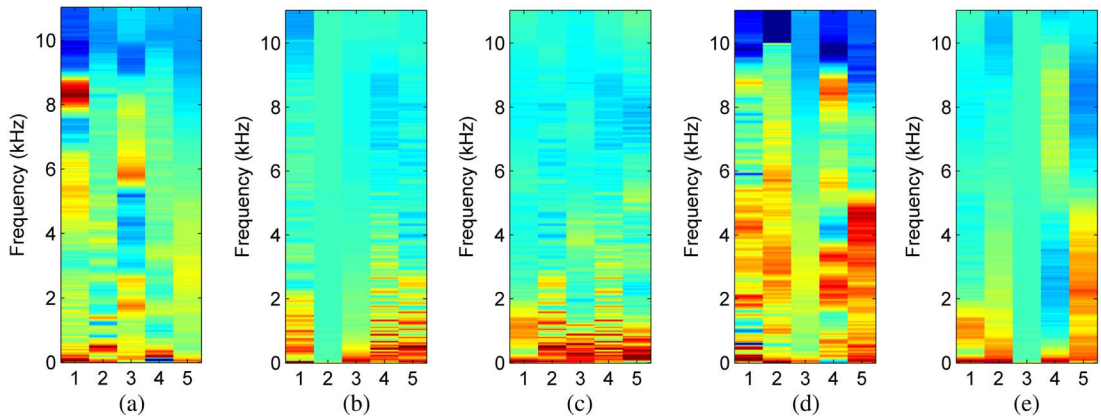

Fig. 2. Top five (from left to right) audio alphabets (first-layer codewords) most frequently used by the five selected genres in the GTZAN data set [25].

B. Preprocessing

Music signals were sampled at 22,050 Hz and cut into 46-ms frames with 50% overlapping, after which the following low-level short-time features were extracted.

- *Log-power spectrogram* (STFT) was obtained by applying logarithmic scaling on the spectrogram. Given our sampling rate and frame size, the dimension $m = 513$.
- *Mel-spectrum* was computed by wrapping the linear-frequency scale into a nonlinear, perceptual-motivated Mel-scale [52] by triangular filters, reducing m to 128.
- *Principal Mel-spectrum components (PMSC)* was a PCA-whitened feature of Mel-spectrum proposed by Hamel *et al.* [21]. In addition to preserving the original dimension of the Mel-spectrum, we consider one more variant that reduces m to 80 by dropping low-variance components.

We did not consider other features such as magnitude spectrogram, MFCC, and CQT for their sub-optimal performance shown in recent BoF-based MIR studies [13], [14], [21].

VI. Experiment Results

Table IV summarizes the algorithms and important parameters we evaluated for the BoF-based feature representation of music. According to the suggestions in [31], we set the default values for the codebook sizes to 1024, and the length of the texture window (for segmentation) to 2.5 seconds. We used the USPOP 2002 data set as the training corpus, which contains over 6,700 30-second clips of pop music.[13] This way, the codebooks used in the three MIR tasks are the same. Although the USPOP data set might not be large enough to cover the universe of music, our evaluation shows that features learnt from it already led to result comparable to state-of-the-art in the three tasks, using just linear kernel SVM for classifier training [62]. For future work, it is straightforward to experiment with a larger training corpus.

We began comparing the performance of SLBoF features obtained from three types of codebooks and five encoding methods. This is followed by comparisons of different pooling methods, power normalization and low-level features. Next, we moved on to the comparisons of different codebooks and pooling methods for the two-layer method DLBoF. The size of first- and second-layer codebooks and the segment length were also studied. Finally, various tf-idf functions and dimension reduction methods are compared based on the DLBoF features. Due to space limit, the evaluation was mostly done for genre classification and instrument recognition. For auto-tagging, we only evaluated the performance of the optimal settings for SLBoF and DLBoF.

[13]http://labrosa.ee.columbia.edu/projects/musicsim/uspop2002.html

A. Evaluation of SLBoF

According to the common evaluation protocol, we used 10-fold cross validation (CV) for genre classification on GTZAN and instrument recognition on the MTG data set [26].

1) Visualization: To provide a sense of the audio alphabets, Fig. 2 shows the top five audio alphabets most frequently used by five of the ten genres in the GTZAN data set. Each audio alphabet is represented in the frequency domain. Please note that the audio alphabets were learnt from USPOP using ODL and then used to encode the clips in GTZAN using L1. It can be seen that different genres show the different preferences of audio codewords. For example, 'blues' and 'metal' prefer codewords that are inharmonic, high-frequency, and large in spectral variance, whereas 'classical' and 'jazz' prefer those with opposite qualities. This is reasonable as classical music is mostly performed with much more pitched instrument than percussion instruments and human voices.

2) Codebook Types & Encoding Methods: Figures 3(a) and 3(b) illustrate the average accuracy in genre classification and predominant instrument recognition for pitched instruments, respectively, using 60 different combinations of experimental settings by codebook generation methods, encoding methods, pooling methods (max or sum pooling) and power normalization (taking square root or not). From the bold lines in both figures, it can be seen that ODL generally performs better than the other two dictionary learning algorithms. As for encoding method, we see that ODL+L1 performs the best for both cases. When sum pooling and power normalization are performed (i.e., SUM+SQRT), it achieves 81.2% for genre classification and 64.7% for predominant instrument recognition. The performance difference between ODL and either $k$means or exemplar dictionaries is significant (p-value < 0.001, d.f. $= 18$) under the two-tailed t-test. Generally speaking, L1 outperforms other

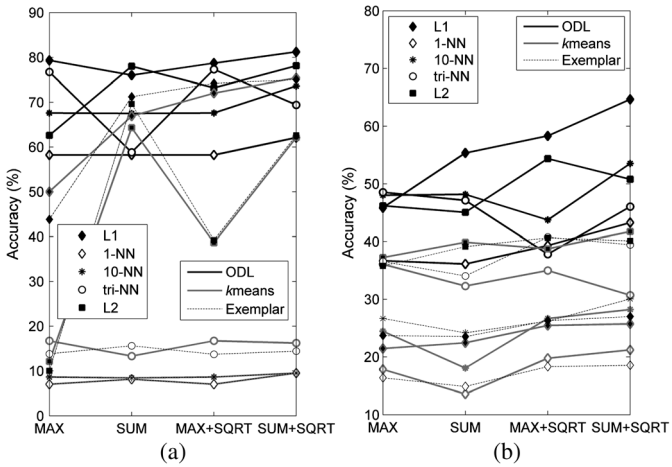

Fig. 3. Average accuracy of GTZAN and MTG datasets using different codebooks (ODL, $k$means, Exemplar), encoding methods (L1, 1-NN, 10-NN, tri-NN, L2), pooling methods (max or sum), and power normalization (SQRT or not power normalized).

encoding methods, regardless of pooling methods and power normalization.

In Fig. 3(a), L1 is the best encoding method for almost all cases. In contrast, in Fig. 3(b), L2 performs better than L1 for $k$means- and exemplar-based codebooks. We note that L2 is the only method that generates dense BoF features among the five encoding methods, but it is only inferior to L1. When SUM+SQRT is applied, ODL+L1 is not significantly better than ODL+L2 for genre classification ($\mathrm{p} = 0.089$, $\mathrm{d.f.} = 18$). It seems that dense BoF features can also be competitive.

3) Pooling Methods & Encoding Methods: From Fig. 3, we can see that power normalization is generally helpful, and that sum pooling usually performs better than max pooling. The combination of SUM+SQRT leads to the best accuracy for either genre classification or instrument recognition.

There seems to be a performance dependency between clip-level pooling and encoding method. For 1-NN, 10-NN and tri-NN, max pooling is generally better, whereas for L1 and L2 max pooling is not always better. For genre classification, max pooling is also effective; if power normalization is not performed, max pooling performs slightly better than sum pooling, for the case of ODL+L1 (79.3% vs. 76.0%). However, for predominant instrument recognition, sum pooling is much better than max pooling; the performance difference between SUM+SQRT and MAX+SQRT is significant ($\mathrm{p} < 0.005$, $\mathrm{d.f.} = 18$) for the case of ODL+L1. When the baseline performance is relatively low (e.g., instrument recognition), it is advisable to use SUM+SQRT. In contrast, when the baseline performance is high (e.g., genre classification), the effect of pooling method and power normalization is less remarkable.[14]

[14]We have also experimented with median pooling [22], which takes the median values of codewords over time. However, results show that median pooling is significantly inferior to sum or max pooling. For example, for ODL+L1, median pooling degrades the average accuracy 51.0% for genre classification. To explain this, we should note that what BoAA features measure is the relative "occurrence" of each audio-alphabets within one clip. If one specific audio-alphabet occurs for only 5 seconds in a 30-second clip, taking median values is likely to eliminate the contribution of this audio-alphabet, whereas max pooling or sum pooling does not.

TABLE V
PERFORMANCE COMPARISON ON GENRE CLASSIFICATION, PITCHED INSTRUMENT RECOGNITION AND PERCUSSION SOUND DETECTION (IN % ACCURACY) FOR DIFFERENT CODEBOOKS AND POOLING METHODS FOR DLBOF

Pooling setting	Genre	Pitched instrument	Percussion instrument	Tagging (AUC)
Null–SUM (SLBoF)	81.2	64.7	89.4	70.8
Null–MAX (SLBoF)	78.7	58.3	64.5	71.3
SUM–SUM (DLBoF)	81.3	67.4	89.5	70.9
MAX–SUM (DLBoF)	81.4	67.8	89.3	71.0
SUM–MAX (DLBoF)	80.4	60.2	64.5	71.4
MAX–MAX (DLBoF)	79.7	60.1	64.5	71.7

TABLE VI
PERFORMANCE COMPARISON ON GENRE CLASSIFICATION (IN % ACCURACY) FOR DIFFERENT TF-IDF FUNCTIONS FOR DLBOF

	idf_b	idf_p	idf_{e1}	idf_{e2}	idf_o	AVG
tf_b	82.0	76.4	81.5	81.2	75.2	79.3
tf_{l1}	81.6	75.9	82.0	82.7	76.2	79.7
tf_{l2}	79.6	71.1	79.7	80.7	71.2	76.5
tf_{o1}	80.9	75.7	82.2	81.9	75.1	79.2
tf_{o2}	80.8	75.6	82.5	81.6	75.5	79.2
AVG	81.0	75.0	81.6	81.6	74.4	78.7

B. Evaluation of DLBoF

We then evaluated the performance of DLBoF for genre classification and predominant instrument recognition (including pitched and percussion), using the same codebook generation methods for both the two layers. We fixed the encoding method to L1 and adopted power normalization. Table V shows the results of different codebook generation methods, coupling with different pooling methods at the segment- and clip- levels. For example, SUM–MAX denotes segment-level sum pooling and clip-level max pooling, whereas null–SUM indicates no pooling at the segment level (i.e., SLBoF). We see that DLBoF greatly improves the accuracy for pitched instrument recognition (from 64.7% to 67.4%); the difference between DLBoF and SLBoF is significant. In contrast, for less challenging tasks such as genre classification or percussion instrument recognition, using one layer is enough.

A closer comparison of the pooling methods reveals that the result is relatively less sensitive to segment-level pooling than to clip-level pooling. Taking sum for clip-level pooling usually leads to better performance, and the result holds for both SLBoF and DLBoF. The most obvious case is percussion instrument, where clip-level max pooling is significantly worse. On average, SUM–SUM pooling performs the best for DLBoF, while MAX-SUM is a competitive alternative.

C. Text-Like Processing for Genre Classification

1) Effect of tf-idf: Table VI shows the result of the 25 variants of tf-idf measure described in Section IV for DLBoF audio words, using ODL+L1 and MAX–SUM pooling.[15] Power normalization was performed after tf-idf processing.

[15]We have exhaustively evaluated the performance of all tf-idf functions when different dictionaries, encoding methods, pooling methods and power normalization methods are used. Results show that ODL+L1 and MAX–SUM are the optimal settings for all tf-idf functions.

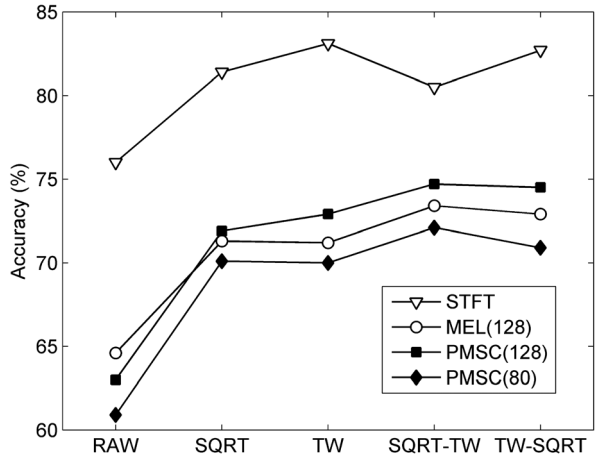

Fig. 4. Performance comparison on genre classification for different post-processing and low-level features of DLBoF.

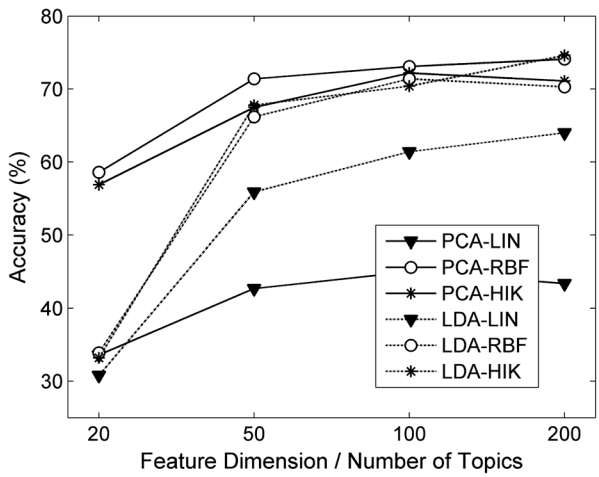

Fig. 5. Performance comparison on genre classification for LDA- and PCA-transformed SLBoF features obtained by ODL+L1, with various feature dimensions (number of topics) and kernel functions for SVM.

The following observations can be made from Table VI. First, the best two results are obtained by $tf_{l1} \cdot idf_{e2}$ (82.7%) and $tf_{l1} \cdot idf_{e1}$ (82.0%), both of which outperform the non-tfidf weighting one (81.4%; cf. Table V). Interestingly, both idf_{e1} and idf_{e2} are kinds of entropy measure function, and tf_{l1} is an alternative logarithm form. Second, the standard formulation $tf_b \cdot idf_b$ is actually competitive as well (82.0%). Third, by observing the average result, we see that in general the choice of the idf function effects the performance (from 74.4% to 81.6%) more than the choice of the tf function (from 76.5% to 79.3%). Fourth, we note that the performance degrades when we use tf_{l2} instead of tf_{l1}, presumably because tf_{l2} results in negative values when $f_{d,t} < 1$, which happens as a result of soft encoding. Similarly, the performance of idf_p and idf_o are poor because they incur negative values when $F_t > 0.5|\mathcal{D}|$ (i.e., having terms that appear in more than half of the corpus documents). Finally, the tf-idf measures adopted by Okapi BM25 do not offer advantages, possibly because we are doing music categorization instead of text-based search, or because the optimal values for the parameters κ and b for text words do not apply as well to audio codewords.[16] Note that there are no significant differences among the most competitive tf-idf functions (such as $tf_{l1} \cdot idf_{e2}$ and $tf_{l1} \cdot idf_{e1}$); the optimal choice of tf-idf function seems to be data-dependent.

In the following study, we adopted $tf_{l1} \cdot idf_{e2}$ as the main term weighting method (denoted as TW).

2) Post-Processing and Low-Level Features: As both power normalization and tf-idf weighting are post-processing operations that are performed after encoding, it is interesting to know which one is more effective and whether it is beneficial to combine them. Fig. 4 shows that, when using log-power spectrogram (STFT) as the feature representation, both tf-idf and power normalization significantly ($p < 0.001$) improve the performance comparing to the no post-processing case (i.e., RAW). Moreover, tf-idf appears to be more effective than power normalization. If we employ tf-idf first and then power normalization (TW–SQRT), the accuracy of genre classification can be improved slightly.

Fig. 4 also shows that STFT outperforms other features by a great margin. Although PMSC performs better than mel-spectrum, it is still significantly inferior to STFT. Interestingly, unlike the case of STFT, SQRT–TW performs better than TW-SQRT for other three features, suggesting that the optimal ordering of applying these two operations is feature-dependent.

3) Latent Dirichlet Allocation: We compared the performance of LDA with principal component analysis (PCA), which is arguably the most popular dimension reduction method. Our hypothesis is that, if audio codewords are more text-like, LDA should perform better than PCA. For this comparison, we experimented with different numbers of latent topics (or the dimensions of the reduced space in the case of PCA) and different kernel functions of SVM, with the intuition that PCA-transformed features may work better for nonlinear kernels such as radial basis function (RBF) kernel and histogram intersection kernel (HIK) [44]. We considered only single-layer encoding and used ODL+L1.

Fig. 5 shows that, however, PCA (the hard lines) generally performs better than LDA (the dash lines) in most cases, especially when the reduced feature space is smaller than 50 in size. LDA outperforms PCA only when using the linear kernel, but it appears that the linear kernel is not competitive to the other two kernels in the reduced feature space.[17] It seems that LDA is not as useful for audio codewords as for text words. Similar observations were made when $k$means was employed for codebook learning.

Fig. 5 also shows that LDA performs well when we used the HIK kernel, which is in particular designed for histogram-like features. In contrast, PCA works the best with the RBF kernel, which implicitly assumes the data to be Gaussian-like. When the feature dimension is 200, both LDA+HIK and PCA+RBF

[16]The latter point is validated through a grid search over $\kappa \in [0, 10]$ and $b \in [0, 1]$, which identifies that the best result is obtained with $\kappa = 6.5$ (higher threshold rate of terms) and $b = 0.3$ (lower document length normalization). However, the accuracy is still inferior to that achieved by $tf_{l1} \cdot idf_{e2}$.

[17]For the original feature space spanned by the audio codewords, Yeh *et al.* [31] showed that linear kernel is only slightly inferior to HIK for BoF-based genre classification on GTZAN. Moreover, linear kernel is much more efficient and scalable.

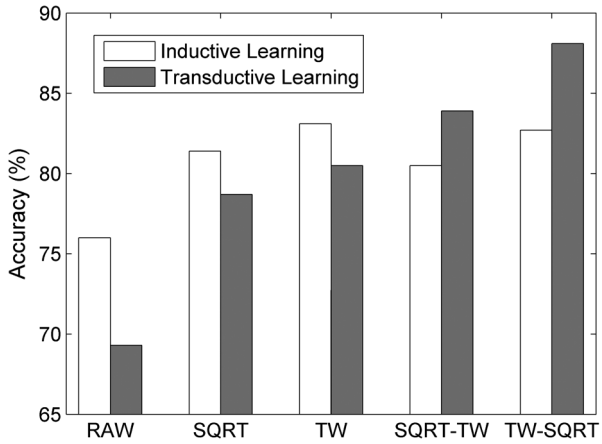

Fig. 6. Performance comparison on genre classification for different post-processing methods under inductive and transductive learning scenarios.

attain 75% accuracy in genre classification, which is close to the performance (76%) of the original 1024-D (set by k_1) feature.

The above result shows that both PCA and LDA reduce the feature dimension greatly without much performance loss, an important merit for the processing of high-dimensional data. However, in terms of accuracy, tf-idf weighting plus power normalization is a preferable post-processing step.

D. Inductive Learning and Transductive Learning

The transductive learning scenario can be tested by generating codebooks from the GTZAN data set itself. Fig. 6 compares the results of the raw features (DLBoF) and those post-processed by power normalization and/or tf-idf functions in the two learning scenarios. We see that in the transductive mode, raw features lead to lower accuracy. However, if power normalization and tf-idf are applied, the accuracy increases considerably. In the case of TW–SQRT, the transductive mode gets 88.1% accuracy, which is much higher than those obtained by the inductive mode. However, this high accuracy does not guarantee generalizability. Moreover, we see that the result is more sensitive to different post-processing operations for the case of transductive learning.

E. Application to Semantic Annotation and Retrieval

For CAL500, we followed the common protocol [27] and reported the result of 5-fold CV. The performance of semantic annotation was evaluated using per-tag precision, recall, F-score, whereas that of retrieval was measured by the AUC measure for each query tag.[18] We used linear SVM and trained binary classifier for each tag independently.

Table VII shows the average result over the 97 tags. It can be seen that both SQRT and TW offer improvement for SLBoF, but not TW–SQRT. In addition, DLBoF significantly outperforms SLBoF for all the four performance measures, showing again that the two-layer structure is effective. The best result (0.269 in F-score and 0.712 in AUC) was obtained by MAX–SUM, which is also effective for genre classification and instrument recognition (cf. Table V).

[18]AUC, or the area under the receiver operating characteristic curve, places more emphasis on the performance of the ranked order of songs, rather than the binary relevance of the retrieved songs.

TABLE VII
PERFORMANCE COMPARISON FOR DIFFERENT SETTINGS ON CAL500

SLBoF	Precision	Recall	F-score	AUC
SUM–RAW	0.434	0.219	0.263	0.702
SUM–SQRT	0.436	0.219	0.264	0.704
SUM–TW	0.436	0.219	0.264	0.704
SUM–TW–SQRT	0.434	0.218	0.260	0.701
DLBoF+TW–SQRT	Precision	Recall	F-score	AUC
SUM–SUM	0.441	0.223	0.267	0.710
MAX–SUM	0.443	0.225	0.269	0.712

F. Comparison with State-of-the-Art

For genre classification, the best result we obtain with DLBoF+TW–SQRT reaches 82.7% in accuracy, which is comparable to existing works that use SC or DBN [6], [10], [13]. If we consider transductive learning, the best accuracy we obtained reaches 88.1%, which is very competitive.

For instrument recognition, the state-of-the-art result presented in [26] is 63% for pitched instruments and 89% for percussion instruments, using RBF kernel SVM. As Table V shows, SLBoF performs slightly better than this prior art, and DLBoF+TW–SQRT with MAX–SUM pooling performs even better (66.7% and 89.4%). Please note that, without tf-idf and square root, we can only get 53.6% for pitched instrument. That is to say, the improvement is due to the two-layer structure and the text-like post-processing techniques.

For auto-tagging, our result (F-score 0.269/ AUC 0.712) is comparable to a recent work [63] (0.264/ 0.723), which can be considered as a hybrid of generative/discriminative method. Although Nam *et al.* reported better result for the same dataset (0.292/ 0.754), they adopted transductive learning. In contrast, we employed inductive learning and used the same dictionary (learnt from USPOP) for three MIR tasks.

VII. DISCUSSION

A. On the Two-Layer Structure & Alphabet-Word Structure

The analogue of first-layer features to "alphabets" and second-layer features to "words" needs more justification. In the present system, audio-alphabets are not utilized directly as the basic elements of audio-words, but the combination coefficients of the audio-alphabets. Audio-alphabets are in the signal space, whereas audio-words are in the indexical space. It might be more reasonable to form an analogy on the scale of the signifying objects, instead of on the content itself.

As our evaluation shows, audio-alphabets can be treated directly as individual terms in tf-idf weighting and LDA, whereas text alphabets cannot in the case of text IR. Moreover, we note that the representations may be complementary to one another, as audio-alphabets are local descriptors for short-time frames, whereas audio-words represent long-term information. For MIR tasks for which the performance of audio-alphabets are good enough (e.g., percussion instrument recognition), adding audio-words does not lead to significant improvement. We have also found that audio-words lead to poor performance when being

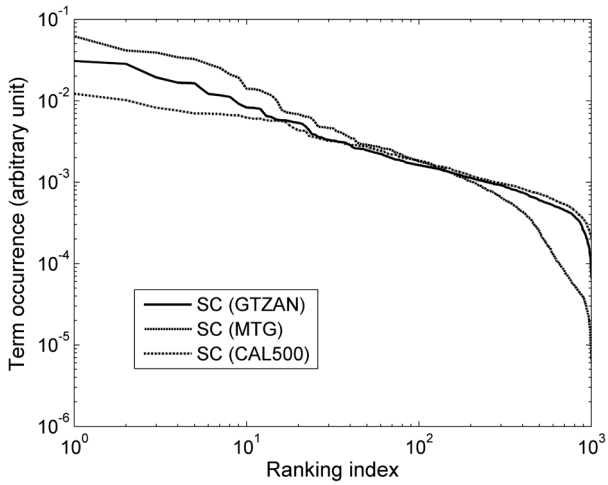

Fig. 7. Log-log rank-frequency distribution of audio-alphabets observed in three data sets. The audio-alphabets are learnt from USPOP using ODL, and are assigned to the signals in these data sets using L1 encoding.

used alone, and show advantages only when being fused with audio-alphabets.

The encoding of audio-words using the second-layer codebook D_2 is a problem about "sparsely-encoding a sparse input feature using a sparse dictionary." Although both D_1 and D_2 generate sparse codes, the former is a dense codebook, whereas the latter is a sparse one. From our evaluation, it seems that classification based on the second-layer codebook is more unstable and task-dependent.

B. On Text-Like Processing & Zipf's Law

According to Zipf's law, a natural statistics of terms would approximate a straight-line when the term occurrance and term rank are ploted in log-log scale [64]. To see if the audio codewords follow the Zipf's law, we rank the 1,024 audio-alphabets in a descending order of the relative term occurrence (normalized by document length) in the GTZAN, MTG and CAL500 data sets and plot the distribution in a log-log scale. As Fig. 7 shows, although the distributions of the three data set appear to be straight line-like, the three differ a lot in the high-ranked parts. For MTG, we observe frequent terms that appear in a majority of the documents (songs), implying higher similarity between the codewords. The distribution of CAL500 is flatter than the other two, implying more uniform distribution of terms. From our data we also observe that the sparsity of CAL500 codewords is the lowest, followed by GTZAN and then MTG. Empirically, we found that tf-idf cannot work well for dense features, and that power normalization is not effective for data with shorter-tail distribution. This explains why TW–SQRT does not improve the result for CAL500.

C. On LDA & Audio Vocabulary

Terms in human language are discrete objects, while audio-alphabets and audio-words are "continuous" objects. The definition of words is relatively "hard," whereas the definition of audio codewords is soft (e.g., according to the loss function in SC and Euclidean distance in VQ). Accordingly, one audio word may correspond to only a part of an audio objects. The difference is crucial in that most of the retrieval models of text IR rely on the *word independence assumption* [23] and consider that a word occurs independently from the others. Moreover, as BoF model are essentially real-valued vectors, for text IR techniques (e.g., LDA) that require integer word counts, we lose information when converting the values from floating points to integers (e.g., by rounding). Due to the above reasons, we find LDA less applicable to audio codewords. In contrast, PCA preservers more information even when the feature dimension is reduced to less than 50, possibly because such rounding operations are not needed.

We also note that, unlike text words, we can obtain lots of different vocabularies for one given music signal using different encoding methods and parameters. We can even learn multiple codebooks from low-level features that characterize different perceptual dimensions of music (e.g., timbre, pitch, and rhythm) and combine the codebooks to form a better audio vocabulary. This is part of our future work.

VIII. Conclusion

In this paper, we have described a two-layer system that integrates recent development in SC and BoF-based feature representation for music. The resultant audio codewords are modeled as symbols in a text corpora, using a hierarchical structure that resembles the alphabet-word-document structure of text. A thorough overview and empirical study on codebook learning, encoding, pooling, power normalization, tf-idf weighting, and LDA is given. Our performance study identifies the optimal setting for BoF-based music classification, including using log-power spectrogram for low-level feature representation, ODL and sparse coding for codebook learning and codeword assignment, max pooling in the segment-level, sum pooling in the clip-level, logarithm tf function plus entropy-based idf function, and square root power normalization. The above setting has been shown useful for three different MIR tasks, including genre classification, predominant instrument recognition, semantic annotation and retrieval. Topic modeling methods such as LDA are found not that useful for BoF features. This work contributes to the understanding of the widely-used BoF representation of music and suggests a novel SC-based multi-level model for MIR problems.

Acknowledgment

The authors would like to thank the anonymous reviewers for valuable comments that improved the quality of this paper.

References

[1] E. C. Smith and M. S. Lewicki, "Efficient auditory coding," *Nature*, vol. 439, no. 7079, pp. 978–982, 2006.
[2] P. Manzagol, T. Bertin-Mahieux, and D. Eck, "On the use of sparse time-relative auditory codes for music," in *Proc. ISMIR*, 2008, pp. 14–18.
[3] R. F. Lyon, M. Rehn, S. Bengio, T. C. Walters, and G. Chechik, "Sound retrieval and ranking using sparse auditory representations," *Neural Computat.*, vol. 20, pp. 2390–2416, 2010.
[4] M. D. Plumbley, T. Blumensath, L. Daudet, R. Gribonval, and M. E. Davies, "Sparse representations in audio and music: From coding to source separation," *Proc. IEEE*, vol. 98, no. 6, pp. 995–1005, 2010.
[5] H. Lee, Y. Largman, P. Pham, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Proc. NIPS*, 2009, pp. 1096–1104.
[6] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks," in *Proc. ISMIR*, 2010, pp. 339–344.

[7] J. Nam, J. Ngiam, H. Lee, and M. Slaney, "A classification-based polyphonic piano transcription approach using learned feature representations," in *Proc. ISMIR*, 2011, pp. 175–180.

[8] S. Scholler and H. Purwins, "Sparse approximations for drum sound classification," *IEEE J. Select. Topics Signal Process.*, vol. 5, no. 5, pp. 933–940, 2011.

[9] S. Dieleman, P. Brakel, and B. Schrauwen, "Audio based music classification with a pre trained convolutional network," in *Proc. ISMIR*, 2011, pp. 669–674.

[10] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun, "Unsupervised learning of sparse features for scalable audio classification," in *Proc. ISMIR*, 2011, pp. 681–686.

[11] E. M. Schmidt, J. Scott, and Y. E. Kim, "Feature learning in dynamic environments: Modeling the acoustic structure of musical emotion," in *Proc. ISMIR*, 2012, pp. 325–330.

[12] P. D. Corey Kereliuk, "Sparse atomic modeling of audio: A review," in *Proc. Int. Conf. Digital Audio Effects*, 2011, pp. 81–92.

[13] C.-C. M. Yeh and Y.-H. Yang, "Supervised dictionary learning for music genre classification," in *Proc. ACM ICMR*, 2012.

[14] J. Nam, J. Herrera, M. Slaney, and J. Smith, "Learning sparse feature representations for music annotation and retrieval," in *Proc. ISMIR*, 2012, pp. 565–560.

[15] E. Battenberg and D. Wessel, "Analyzing drum patterns using conditional deep belief networks," in *Proc. ISMIR*, 2012, pp. 37–42.

[16] C.-T. Lee, Y.-H. Yang, and H. H. Chen, "Multipitch estimation of piano music by exemplar-based sparse representation," *IEEE Trans. Multimedia*, to be published.

[17] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Deep architectures and automatic feature learning in music informatics," in *Proc. ISMIR*, 2012, pp. 403–408.

[18] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. ICML*, 2009, pp. 689–696.

[19] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[20] J.-J. Aucouturier, B. Defreville, and F. Pachet, "The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music," *J. Acoust. Soc. Amer.*, vol. 122, no. 2, pp. 881–891, 2007.

[21] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck, "Temporal pooling and multiscale learning for automatic annotation and ranking of music audio," in *Proc. ISMIR*, 2011, pp. 729–734.

[22] E. Pampalk, A. Rauber, and D. Merkl, "Content-based organization and visualization of music archives," in *Proc. 10th ACM Int. Conf. Multimedia (MULTIMEDIA '02)*, New York, NY, USA, 2002, pp. 570–579 [Online]. Available: http://doi.acm.org/10.1145/641007.641121

[23] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Reading, MA, USA: Addison-Wesley, 1999.

[24] A. Berenzweig, B. Logan, D. P. W. Ellis, and B. Whitman, "A large-scale evaluation of acoustic and subjective music similarity measures," *Comput. Music J.*, vol. 28, no. 2, pp. 63–76, 2004.

[25] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 5, pp. 293–302, 2002.

[26] F. Fuhrmann, "Automatic musical instrument recognition from polyphonic music audio signals," Ph.D. dissertation, Univ. Pompeu Fabra, Barcelona, Spain, 2012.

[27] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Towards musical query-by-semantic-description using the CAL500 data set," in *Proc. ACM SIGIR*, 2007, pp. 439–446.

[28] M. Riley, E. Heinen, and J. Ghosh, "A text retrieval approach to content-based audio retrieval," in *Proc. ISMIR*, 2008, pp. 295–300.

[29] K. Aryafar and A. Shokoufandeh, "Music genre classification using explicit semantic analysis," in *Proc. ACM Workshop Music Information Retrieval With User-Centered and Multimodal Strategies*, 2011, pp. 33–37.

[30] Z. Fu, G. Lu, K.-M. Ting, and D. Zhang, "Music classification via the bag-of-features approach," *Pattern Recognit. Lett.*, vol. 32, pp. 1768–1777, 2011.

[31] C.-C. M. Yeh, L. Su, and Y.-H. Yang, "Dual-layer bag-of-frames model for music genre classification," in *Proc. IEEE ICASSP*, 2013.

[32] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[33] K. Seyerlehner, G. Widmer, and P. Knees, "Frame level audio similarity: A codebook approach," in *Proc. Int. Conf. Digital Audio Effects*, 2008.

[34] J. Wülfing and M. Riedmiller, "Unsupervised learning of local features for music classification," in *Proc. ISMIR*, 2012, pp. 139–144.

[35] B. McFee, L. Barrington, and G. R. G. Lanckriet, "Learning content similarity for music recommendation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 8, 2012.

[36] R. J. Weiss and J. P. Bello, "Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization," in *Proc. ISMIR*, 2011, pp. 123–128.

[37] J.-C. Wang, H.-S. Lee, H.-M. Wang, and S.-K. Jeng, "Learning the similarity of audio music in bag-of-frames representation from tagged music data," in *Proc. ISMIR*, 2011.

[38] J. Schlüter and C. Osendorfer, "Music similarity estimation with the mean-covariance restricted Boltzmann machine," in *Proc. Int. Conf. Machine Learning and Applications*, 2011, pp. 118–123.

[39] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA, USA: Kluwer, 1991.

[40] A. Lacoste and D. Eck, "A supervised classification algorithm for note onset detection," *EURASIP J. Adv. Signal Process.*, pp. 1–14, 2007.

[41] E. J. Humphrey, T. Cho, and J. P. Bello, "Learning a robust tonnetzspace transform for automatic chord recognition," in *Proc. IEEE ICASSP*, 2012, pp. 453–456.

[42] A. Coates and A. Ng, "The importance of encoding versus training with sparse coding and vector quantization," in *Proc. ICML*, 2011, pp. 921–928.

[43] B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Statist.*, vol. 32, pp. 407–499, 2004.

[44] S. Maji, A. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," in *Proc. IEEE CVPR*, 2008, pp. 1–8.

[45] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proc. IEEE*, vol. 98, no. 6, pp. 1031–1044, 2010.

[46] J. F. Gemmeke, T. Virtanen, and A. Hurmalainen, "Exemplar-based sparse representations for noise robust automatic speech recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 7, pp. 2067–2080, Sep. 2011.

[47] D. Sculley, "Web-scale k-means clustering," in *Proc. ACM WWW*, 2010, pp. 1177–1178.

[48] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[49] S. S. Chen, D. L. Donoho, Michael, and A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, pp. 33–61, 1998.

[50] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal Statist. Soc.*, vol. 58, pp. 267–288, 1996.

[51] A. Coates, H. Lee, and A. Ng, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. AISTATS*, 2011.

[52] M. Müller, D. P. W. Ellis, A. Klapuri, and G. Richard, "Signal processing for music analysis," *J. Select. Topics Signal Process.*, vol. 5, no. 6, pp. 1088–1110, 2011.

[53] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. IEEE CVPR*, 2009, pp. 1794–1801.

[54] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck, "Temporal pooling and multiscale learning for automatic annotation and ranking of music audio," in *Proc. ISMIR*, 2011, pp. 729–734.

[55] Y.-L. Boureau, J. Ponce, and Y. Lecun, "A theoretical analysis of feature pooling in visual recognition," in *Proc. ICML*, 2010.

[56] S. Robertson, "Understanding inverse document frequency: On theoretical arguments for IDF," *J. Document.*, vol. 60, no. 5, pp. 503–520, 2004.

[57] M. Schedl, T. Pohle, P. Knees, and G. Widmer, "Exploring the music similarity space on the web," *ACM Trans. Inf. Syst.*, vol. 29, no. 3, pp. 14:1–14:24, 2011.

[58] S. E. Robertson, S. Walker, M. M. Beaulieu, M. Gatford, and A. Payne, "Okapi at trec-4," in *Proc. Text Retrieval Conf.*, 1995, pp. 73–96.

[59] M. Lan, C. L. Tan, J. Su, and Y. Lu, "Supervised and traditional term weighting methods for automatic text categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, pp. 721–735, 2009.

[60] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, pp. 1704–1716, 2012.

[61] B. L. Sturm, "An analysis of the GTZAN music genre dataset," in *Proc. ACM Workshop Music Information Retrieval With User-Centered and Multimodal Strategies*, 2012, pp. 7–12.

[62] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, 2001.

[63] E. Coviello, Y. Vaizman, A. B. Chan, and G. R. Lanckriet, "Multivariate autoregressive mixture models for music auto-tagging," in *Proc. ISMIR*, 2012, pp. 547–552.

[64] M. Haro, P. H. Joan Serrà, and Á Corral, "Zipf's law in short-time timbral codings of speech, music, and environmental sound signals," *PLoS ONE*, vol. 7, no. 3, 2012, e33993.

Li Su (S'08–M'13) received the Ph.D. degree in Communication Engineering from National Taiwan University, Taiwan, in 2012. Since May 2012, he has been a Postdoctoral Research Fellow in Research Center for Information Technology Innovation, Academia Sinica, Taiwan. His research interests include music information retrieval, machine learning, signal processing and filter design.

Chin-Chia Michael Yeh received the M.S. degree in Mechanical Engineering from University of California, Los Angeles, in 2011. Since July 2011, he has been a Research Assistant at the Academia Sinica Research Center for Information Technology Innovation. His research interests include music information retrieval and machine learning.

Jen-Yu Liu received the M.S. degree in Computer Vision and Artificial Intelligence from Universitat Autonoma de Barcelona, Spain, in 2011. Since October 2011, he has been a Research Assistant at the Academia Sinica Research Center for Information Technology Innovation. His research interests include music information retrieval, user-centered data mining, and machine learning.

Ju-Chiang Wang received the Ph.D. degree in Electrical Engineering, National Taiwan University, Taiwan, in 2013. He joined Institute of Information Science, Academia Sinica, as a Research Assistant in 2007 and has been a Postdoctoral Research Fellow since February 2013. His research interests mainly encompass music information retrieval, machine learning, and audio signal processing. He won the first prize of Multimedia Grand Challenge in ACM Multimedia Conference 2012 and was awarded the Best Presentation Award of the Sixth Beijing-Hong Kong International Doctoral Forum in 2011.

Yi-Hsuan Yang (M'11) received the Ph.D. degree in Communication Engineering from National Taiwan University, Taiwan, in 2010. Since Sept. 2011, he has been with the Academia Sinica Research Center for Information Technology Innovation, where he is an Assistant Research Fellow. His research interests include music information retrieval, multimedia signal processing, machine learning, and affective computing. He was awarded the IEEE Signal Processing Society (SPS) Young Author Best Paper Award in 2011 and the ACM Multimedia Grand Challenge First Prize in 2012. He is an author of the book Music Emotion Recognition, published by CRC Press in 2011, and a tutorial speaker on music affect recognition in the International Society for Music Information Retrieval Conference (ISMIR) in 2012. He is a co-organizer for the International Workshop on Affective Analysis in Multimedia and the MediaEval Affect Task: Music Affective Analysis, both in 2013.