Supervised Dictionary Learning for Music Genre Classification

Chin-Chia Michael Yeh
Research Center for IT Innovation
Academia Sinica, Taipei, Taiwan
mcyeh@citi.sinica.edu.tw

Yi-Hsuan Yang
Research Center for IT Innovation
Academia Sinica, Taipei, Taiwan
yang@citi.sinica.edu.tw

ABSTRACT

This paper concerns the development of a music codebook for summarizing local feature descriptors computed over time. Comparing to a holistic representation, this text-like representation better captures the rich and time-varying information of music. We systematically compare a number of existing codebook generation techniques and also propose a new one that incorporates labeled data in the dictionary learning process. Several aspects of the encoding system such as local feature extraction and codeword encoding are also analyzed. Our result demonstrates the superiority of sparsity-enforced dictionary learning over conventional VQ-based or exemplar-based methods. With the new supervised dictionary learning algorithm and the optimal settings inferred from the performance study, we achieve state-of-the-art accuracy of music genre classification using just the log-power spectrogram as the local feature descriptor. The classification accuracies for benchmark datasets GTZAN and ISMIR2004Genre are 84.7% and 90.8%, respectively.

Categories and Subject Descriptors

H.5.5 [**Sound and Music Computing**]: Methodologies and techniques, Systems

General Terms

Algorithms, Performance

Keywords

Sparse coding, dictionary learning, genre classification

1. INTRODUCTION

Music is one of the most popular types of multimedia information. The explosive growth of online music streaming and download services and the availability of large storage at low cost have greatly changed our way of listening to and consuming music. It is not unusual that personal collections of music exceed the practical limits on the time we have to listen to them [10]. To help users navigate, search, and organize music information in a more efficient way, an increasing number of content-based music information retrieval (MIR) systems have been developed [6, 33]. The video sharing website Youtube can now detect copyrighted music material that was uploaded to the site, and the Apple iTunes' "Genius" service can automatically generate a playlist of songs from the user's library that go great together.[1][2] New music apps for mobile devices are continually being designed.

Underlying almost all the current MIR systems is the application of signal processing and machine learning techniques to extract meaningful semantic information from music signals. Despite that music research is junior in comparison to the large and mature fields of speech and image/video research, recent years have witnessed a growing number of new algorithms that exploit the particular properties of music, such as its harmonic and rhythmic structures. For example, to analyze the repetitive patterns in a music piece a transposition-invariant similarity measure has been found useful as musical parts may be repeated in another key [39]; to identify the cover versions of a song beat-synchronous modeling is often employed to overcome the variability in tempo of different covers [5]. Although we are still far from a complete representation of the musical features that human are able to compute to perceive and enjoy music, remarkable progress is being made, as reported in a recent survey [33].

An important aspect of music signal processing is the aggregation of audio features computed over time. One may compute features from a sequence of overlapping short-time frames and then take the mean and variance along the temporal dimension to create a single feature vector at a larger time scale [14, 32]. However, such temporal pooling approaches cannot well represent music information that happens in a short temporal moment (e.g., "guitar solo") [30]. As a result, prevalent in current music classification systems is the "bag-of-frames" (BOF) model that represents each music piece as a histogram over a dictionary of music "codewords" selected or learnt from a music collection [20, 22, 27, 30, 31, 50]. This general strategy has been particularly successful in both the semantic annotation of music [22, 30, 50] and image/video [54, 56].

This paper presents an endeavor that aims at improving the BOF model by adapting the recent advances in dictionary learning and sparse coding to MIR.[3] It has been shown

ICMR '12, June 5-8, Hong Kong, China

[1] `http://www.youtube.com`
[2] `http://www.apple.com/itunes/`
[3] A signal is sparse when most of its elements are zero.

that modeling a signal as sparse linear combinations of basis elements in the dictionary is very effective in many signal processing applications [15], and that learning the dictionary instead of using off-the-shelf bases leads to better performance [28]. To investigate how this technique best helps the designer of an MIR system, we are motivated to conduct a comprehensive evaluation that empirically compare different codebook generation methods, from conventional $k$means-based vector quantization method [31] to recent exemplar-based or optimization-based ones [28, 52], and different design choices covering encoding method, local feature description, frequency bands that are taken into account, the size and the non-negativity of the codebook. Several new insights are drawn from the experimental result.

Furthermore, we propose a new algorithm that incorporates ground truth labels in the dictionary learning procedure to enhance the discriminative power of the learnt codewords. Based on this *supervised dictionary learning* (SDL) algorithm, we develop a dictionary-based music genre classification system that obtains state-of-the-art accuracies on two benchmark datasets GTZAN and ISMIR2004Genre using just the log-power spectrogram computed by short-time Fourier transform as the local feature descriptor.

The paper is organized as follows: First, we briefly review related work in Section 2. Next, we introduce the concept of dictionary learning in Section 3 and then describe SDL in Section 4. Section 5 provides the details of a dictionary-based music genre classification system we utilize for systematic evaluation, and Section 6 reports and discusses the result. Finally, we conclude the paper in Section 7.

2. RELATED WORK

Music unfolds over time, and music's most expressive qualities probably relate to its structural changes across time [12]. To take the temporal dynamics of music into account, many short-time local descriptors of music have been developed. Typically a music signal is broken into small, possibly overlapping frames (e.g., 46 ms) that are processed and subject to feature extraction [33]. This leads to a variable-length feature sequence for each signal. To train a classifier, Turnbull *et al.* modeled the feature distribution of each music class by a Gaussian mixture model (GMM) and used the posterior likelihood to determine the association between a test signal and each class [48], whereas Hamel *et al.* investigated temporal pooling strategies that map a sequence of features into a fixed-size feature vector that can be fed to algorithms such as decision tree or support vector machine (SVM) [14]. A multivariate autoregressive model has also been developed to model temporal feature correlation [32].

The short-time descriptors of music can also be quantized or clustered to form *music codewords* that are analogous to words in a text document, with each word corresponding to a semantic element of the audio document [27]. Each song can be represented as a histogram over a dictionary of music codewords and subject to algorithms such as latent topic analysis or text-based keyword search well developed in text research. Such text-like representation of multimedia content has been found extremely useful for visual analysis since the advent of the SIFT (scale invariant feature transform) local interest point descriptor [26] and the bag-of-visterms (or *visual words*) model.

A number of algorithms have been proposed to generate the codewords for music. For instance, McFee *et al.* employed $k$means to cluster a collection of frame-level MFCC vectors and used the cluster centers for vector quantization (VQ) [31]. The codeword histogram of a song is constructed by counting the frequency with which each codeword quantizes the bag of MFCC vectors of that song. Wang *et al.* proposed a framework that models the distribution of a collection of short-time features using GMM and regarded each mixture component as a codeword [50]. The posterior probability of each mixture component yields a soft-assigned encoding criterion that enhances the modeling ability of the GMM-based encoding system over the VQ-based one. Levy *et al.* represented music by a joint vocabulary consisting of both conventional words drawn from social tags and audio codewords generated by using self-organizing map, a topology-preserving clustering algorithm [22].

Sparse representations have also been exploited for MIR applications, mostly for source separation and melody transcription. Many researchers have reported success when decomposing music signals using non-negative matrix factorization or sparse coding methods, as surveyed in [40]. Some approaches used off-the-shelf bases such as modified discrete cosine transform or wavelet basis, while others learned a dictionary from the data with prior assumptions such as a source-filter model or a smoothness constraint [4]. In addition, Lee *et al.* proposed a dictionary-based method for multipitch estimation of polyphonic music, where each codeword corresponds to an *exemplar* of a different parts of a note (attack, sustain, and release) [21]. Henaff *et al.* achieved state-of-the-art accuracy in music genre classification by using an efficient approximated sparse coding method with predictive sparse decomposition [16]. In this paper we also evaluate on genre classification and obtain even better performance than this prior art among others.

To our best knowledge, few attempts if any have been made to systematically compare different codebook generation methods for MIR. We opt for evaluating on genre classification for it is one of the most well studied MIR problems [10]. Nevertheless, the proposed framework is simple to implement and readily applicable to other tasks such as music auto-tagging and content-based music retrieval [6].

3. DICTIONARY LEARNING

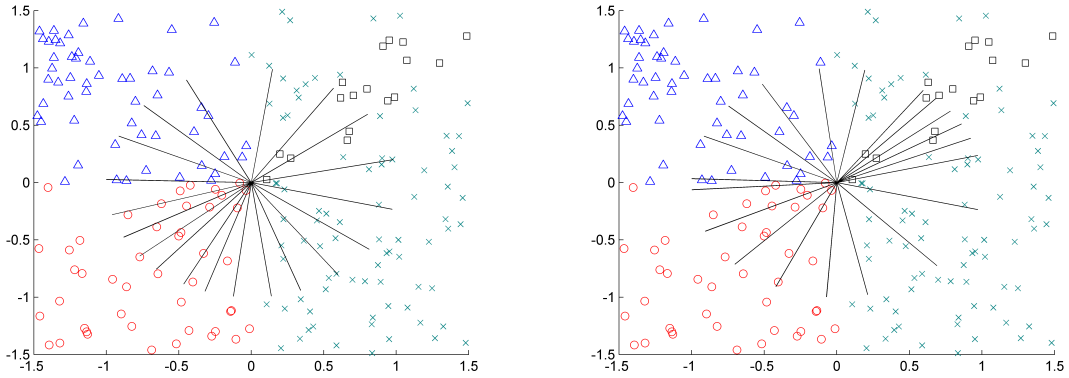
Given an input signal vector $x \in \Re^m$, the sparse representation problem can be mathematically formulated as

$$\alpha^* = \underset{\alpha}{\operatorname{argmin}} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda\|\alpha\|_1, \qquad (1)$$

where $\alpha \in \Re^k$ is a sparse coding of x, $D \in \Re^{m\times k}$ is a given codebook, and λ is a parameter for the trade-off between α's sparsity and the representation accuracy. Typically λ is set to $1/\sqrt{m}$ because that is the classical normalization factor [28], where m is the feature dimension of x. This problem is usually referred to as *basis pursuit* [8] or *Lasso* [47] in the machine learning and statistics literature. It can be solved efficiently by off-the-shelf programs such as LARS-lasso [9].

It has been shown that using a learnt codebook instead of a predefined one improves the performance of sparse coding as the learnt one is more adapt to the data being processed [28]. The codebook learning problem can be formulated as

$$D^* = \underset{D\in\mathcal{C}}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\|x_i - D\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1\right), \qquad (2)$$



(a) Unsupervised codebook.

(b) Supervised codebook.

Figure 1: A scatter plot of a randomly generated two-dimension synthetic data. The black lines represent the learnt codeword vector by dictionary learning [28]. The supervised one learns more codewords for the underrepresented label and therefore has better discriminative power over its unsupervised counterpart.

where x_i denotes the i-th signal among a dataset of n signals, and $\mathcal{C}$ is a set of convex matrices in which the l_2 norm of each column d_j is less than or equal to one, i.e.,

$$\mathcal{C} \triangleq \{D \in \Re^{m \times k} \text{ s.t. } d_j^T d_j \leq 1, \forall j = 1, \ldots, k, \}. \quad (3)$$

This constraint is imposed to constrain the energy of the codewords. The formulation in Eq. 2 is a joint optimization problem in α and D, and a natural solution is to optimize the two variables in an alternative fashion: minimize one variable while keeping the other fixed. Several optimization steps are made until convergence.

Mairal *et al.* proposed a first-order stochastic gradient descent algorithm to solve this joint optimization problem [28]. This algorithm, which is referred to as *online dictionary learning* (ODL), scans through the training set and processes one randomly selected element x_t at a time by alternating a sparse coding step for computing the decomposition α_t of x_t over the dictionary D_{t-1} obtained at the previous iteration, with a dictionary update step that solves the following minimization problem,

$$\begin{aligned} D_t &\triangleq \underset{D \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{t} \sum_{i=1}^{t} \left(\frac{1}{2} \|x_i - D\alpha_i\|_2^2 + \lambda \|\alpha_i\|_1 \right), \\ &= \underset{D \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{t} \left(\frac{1}{2} \operatorname{Tr}(D^T D A_t) - \operatorname{Tr}(D^T B_t) \right), \end{aligned} \quad (4)$$

where $A_t = A_{t-1} + \alpha_t \alpha_t^T$ and $B_t = B_{t-1} + x_t \alpha_t^T$. D_t can be obtained efficiently by using D_{t-1} as warm restart because the cost function defined in Eq. 4 aggregates the past information computed during the previous steps of the algorithm. Because of its low memory consumption and relatively lower computational cost, ODL is more scalable than standard second-order batch algorithms [28]. Therefore, we also adopt ODL for codebook generation in our evaluation.

4. SUPERVISED DICTIONARY LEARNING

As ODL does not consider any labeled information, the generated codewords may not equally span the space of each music label. For classification problems, an unbalanced codebook is not favorable as the examples of an underrepresented class may be encoded with codewords that are associated with nearby classes. As a result, the classifier cannot make accurate prediction as the sparse representation of the underrepresented class is similar to that of other classes.

To address this problem, we improve ODL by exploiting the class labels. Specifically, we develop a two-layer structure that decomposes the codebook into a number of *sub-codebooks*, each of which is trained independently by ODL using the examples of a specific class. For a classification problem with c labels, the codebook is structured as

$$D \triangleq [D_1, D_2, \ldots, D_c], \quad (5)$$

where $D_1, D_2, \ldots, D_c$ are the sub-codebooks for each label, and each sub-codebook has the same size. As we do not know the class membership of an input test signal in prior, we use the combined codebook D to encode it. It is expected that the codewords assigned to the signal are mostly associated with the correct class.

An illustration of the unbalanced codebook problem is shown in Fig. 1 using synthetic data randomly generated in 2D. It can be found that the supervised approach learns more codewords for the underrepresented class and thereby better represents that class than its unsupervised counterpart.

For the encoding system to select codewords that are associated with the correct label for an input signal, we find it useful to impose a non-negativity constraint $\alpha \geq 0$ in both the codebook generation and codeword assignment processes. With this constraint the generated codewords are less likely to span spaces in the opposite direction of the true one, which may happen in the unconstrained case as a result of minimizing $\|x - D\alpha\|_2$, and the encoded codewords for a signal are more likely associated with the true class.

5. DICTIONARY-BASED FRAMEWORK

A system diagram of the dictionary-based music classification framework utilized in this study is presented at Fig. 2, with the training part shaded. It takes as inputs a collection of low-level audio descriptors of training songs and the

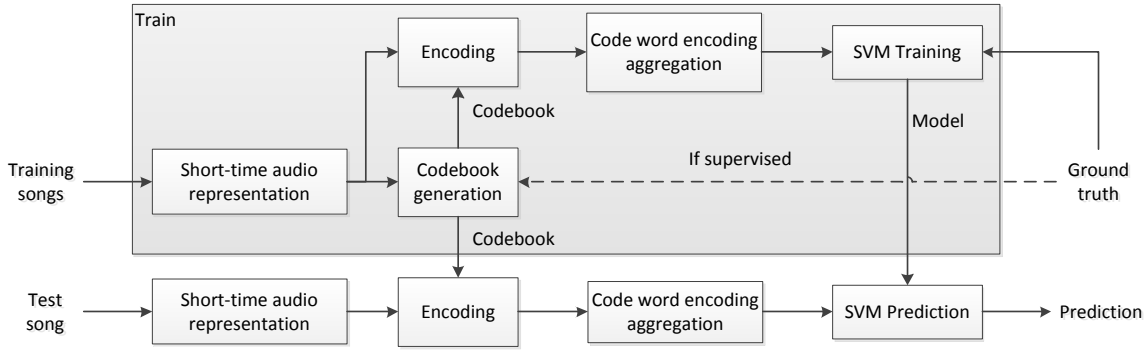


Figure 2: A system diagram of dictionary-based music classification.

corresponding ground truth labels, and generates as outputs a codebook and a SVM model [43] that is trained using the histograms over the codewords as features. The learnt codebook is used to compute the encoding of a test song, which is then fed into the SVM model for classification. The details of each system component is described below.

5.1 Short-time Audio Representation

A great many short-time audio representations have been proposed in the literature, with the magnitude spectrogram computed by short-time Fourier transform possibly being the most fundamental one [33]. It describes the time-varying energy across different frequency bands in a linear frequency scale of the signal. A log-power spectrum is often used because human sensation of power is logarithmic. Moreover, we also consider the following variations,

- **Mel-spectrogram** that is computed by wrapping the linear-frequency scale into a nonlinear Mel-scale by triangular filters. The Mel-scale is designed to approximate the frequency resolution of human ear, which is more sensitive to differences at low frequencies.
- **MFCC** that encodes the coarse shape of the Mel-spectrum by taking the discrete cosine transform. Typically only the 10–20 lowest coefficients are retained and the rest are discarded in order to make the timbre features invariant to pitch information present in the higher coefficients [33]. MFCC is by far the most popular feature representation for music classification [3].
- **Sonogram** which employs techniques such as outer-ear model, Bark-scale critical-bands, and spectral masking to better respect human loudness sensation [34].
- **Constant-Q transform** that replaces linear frequency scale by a logarithmic one to respect the "octave equivalence" of music perception [33, 44], i.e., each doubling in frequency corresponds to an equal musical interval.

In addition, we also consider limiting the range of frequencies over which the above representations is calculated because it has been found that human pitch perception is most strongly influenced by harmonics that occur in a "dominance region" between about 400 and 2,000 Hz [33], and that the frequency range for bass instruments, which carries particular information such as rhythm in a music piece, are between 50 and 400 Hz. Specifically, we consider the following settings: a) use only 50–2,000 Hz, b) use only 400—2,000 Hz, c) use all available frequencies, d) build two classifiers, one for 50–400 Hz and the other for 400-2,000 Hz, and fuse their predictions based on the probability estimates of SVM [43].

We use the MIRtoolbox to compute spectrum, Mel-spectrum, and MFCC, and the MA toolbox to extract sonogram.[4][5] The frame size and hop size are by default set to 1,024 sample and 50% of the frame size, respectively. For constant-Q transform we use the CQT toolbox[6] with 96 filters spanning four octaves from C_2 to C_6. The feature vector of each frame is normalized to a unit l_2-norm vector.

5.2 Codebook Generation

The following codebook generation methods are considered:

- **$k$means** generates a codebook by grouping the training data into k clusters according to l_2 distance, with each cluster center corresponding to a codeword. We regard $k$means as the baseline as it is by far the most common codebook generation method in the literature [41, 51, 53]. Since the amount of training data is usually huge (e.g., for each song there are thousands of frame-level feature vectors), for scalability we adopt the mini-batch $k$means algorithm for clustering [45].
- **ODL**: The online dictionary learning algorithm [28]. While $k$means can be thought as adapting the codebook to the training data for the l_2 distance encoder, ODL adapts the codebook to training data for sparse coding. Due to the consideration of sparse representation, ODL is potentially powerful than $k$means, but its computational cost is relatively higher [28]. Note that ODL does not use a non-negativity constraint.
- **SDL**: The proposed supervised dictionary learning algorithm described in Section 4. Hypothetically it outperforms ODL for supervised tasks. As a slight abuse of terminology, we use SDL♯ to indicate the version without using non-negativity constraint.
- **Exemplar**-based method directly uses all or a subset of the training data as codewords to construct a dictionary. It has been shown useful for audio tasks such as music genre classification [37] and automatic speech

[4] `http://www.jyu.fi/music/coe/materials/mirtoolbox`
[5] `http://www.ofai.at/~elias.pampalk/ma/index.html`
[6] `http://www.elec.qmul.ac.uk/people/anssik/cqt`

recognition [11]. Exemplar-based method is conceptually opposite to the previous ones as it does not adapt the codebook for encoding. Its computational cost is low as no learning is needed.

5.3 Codeword Encoding

We consider the following two encoding methods:

- **L2**-based encoding, or vector quantization, is perhaps the most common way for codeword encoding [41, 51, 53]. It encodes a given signal x by solving the following constrained minimization problem,

$$\alpha^* = \underset{\|\alpha\|_0=1}{\operatorname{argmin}} \|x - D\alpha\|_2, \quad (6)$$

  where $\|\cdot\|_0$ denotes the zero norm, or the number of nonzero elements. In other words, only one codeword that is closest to the signal is selected for encoding.

- **L1**-based encoding obtains a sparse coding α of x by solving Eq. 1. It can select multiple, but just a few, codewords for encoding and assign a membership $\alpha_k \in [0, 1]$ for each selected codeword. We use LARS-lasso [9] to achieve L1-based encoding for it has a C-based implementation that is efficient and publicly available.

Different combinations of codebook generator and codeword encoder lead to different encoding systems. Some combinations correspond to existing methods, e.g., $k$means+L2 [31], ODL+L1 [28], and Exemplar+L1 [21], while others are essentially new algorithms. For example, $k$means+L1 uses the cluster centers computed by $k$means for sparse coding, whereas ODL+L2 picks the codeword with largest α_k from the learnt codebook to encode a signal. Section 6.2 presents an empirical comparison of all the possible combinations.

5.4 Bag-of-Histograms Encoding Aggregation

For classification we require a song-level representation of each song. A typical approach is to construct a histogram that accumulates the frequency of occurrence of each codeword over the short-time frames [41, 51]. This method, albeit simple and effective, is not optimal. It has been found that partitioning a song into short segments, each span a number of frames, usually improves the classification accuracy [2, 49]. These segments are called "texture windows" by Tzanetakis *et al.* as it should correspond to the minimum time amount of music that is necessary to identify a particular music "texture" [49]. It has been found that the optimal window size for the texture window falls between 3–5 seconds [13]. Therefore, we aggregate the codeword encoding over a texture window of 5 seconds, with 50% overlap, and represent a song as a *bag-of-histograms*.

As for training, our system uses all the histograms from the training songs as independent training instances. The label of each histogram is inherited from the associated song. As for prediction, we predict the probability estimate [43] of the association between each texture window and each label, and aggregate the probability estimates over the bag-of-histograms with summation. In the end, the label with the maximum probability is selected as the final prediction.

5.5 Histogram Intersection Kernel and SVM

The histogram intersection kernel (HIK), $K_{HI}(h_a, h_b) = \sum_{j=1}^{k} \min(h_a(j), h_b(j))$ is often used a measurement of similarity between histograms h_a and h_b, and because it is positive definite it can be used as a kernel for SVM [29,43]. It has been shown that for histogram features, the use of HIK for SVM usually outperforms linear of nonlinear kernels such as the radial basis function [29]. Moreover, HIK SVM is computationally comparable with linear SVM and therefore is scalable to large data. Consequently, we use HIK SVM as the classification algorithm in our system.

Table 1: Classification accuracy of different local feature descriptors for GTZAN.

	Dimension	$k$means+L2	ODL+L1
Spectrogram	513	65.6%	82.6%
Mel-spctrogram	40	65.5%	71.7%
MFCC	20	69.1%	73.7%
Sonogram	23	68.3%	72.8%
CQT	96	65.2%	74.5%

6. EXPERIMENT

We evaluate the performance of the dictionary-based framework on music genre classification, one of the most well studied problems in MIR [10]. Two benchmark datasets are employed. The first dataset GTZAN is composed of 1,000 30-second clips covering ten genres [49], whereas the second dataset ISMIR2004Genre consists of 1,458 full-length songs covering six genres.[7][8] While GTZAN is a balanced dataset, with 100 clips per genre, ISMIR2004Genre is unbalanced (e.g., 320 examples for classical but 26 for jazz_blues). Each song is converted to a standard mono-channel and 22,050 Hz sampling rate WAV format.

Evaluation on GTZAN is typically conducted using a stratified 10-fold cross validation, with the class distribution in each fold balanced [7, 16, 38]. ISMIR2004Genre comes with predefined training and development half-half split, so one uses the development set for testing [38]. The performance is measured in terms of the (average) classification accuracy.

6.1 Comparison of Local Feature Descriptors

We first compare the performance of different local feature descriptors for classifying GTZAN using the existing methods $k$means+L2 and ODL+L1 as the encoding methods. The codebook size is set to 500. We show the results in Table 1. It can be found that ODL+L1 consistently outperforms $k$means+L2 regardless of the feature representation, showing that sparse coding is more effective than VQ.

For $k$means+L2, using MFCC leads to the best accuracy 69.1%, which is slightly better than the 68.3% achieved by using sonogram. Interestingly, $k$means+L2+MFCC corresponds to the setting utilized in prior works [31]. It seems that the classification accuracy of $k$means+L2 is inversely proportional to the dimension of the adopted feature representation. Using a high-dimensional feature representation may incur the *curse-of-dimensionality* problem [43] for a l_2-based method like VQ and thereby degrade performance. As

[7]The datasets are available at `http://opihi.cs.uvic.ca/sound/genres.tar.gz` and `http://ismir2004.ismir.net/genre_contest/index.htm`.

[8]The genre classes of GTZAN is classical, country, disco, hiphop, jazz, rock, blues, reggae, pop, and metal, whereas those of ISMIR2004Genre is classical, electronic, jazz_blues, metal_punk, rock_pop, and world.

Table 2: Classification accuracy of different encoding methods for GTZAN.

	MFCC				Spectrogram				
	*k*means	Exemplar	ODL	SDL	*k*means	Exemplar	ODL	SDL♯	SDL
L2 (VQ)	70.1%	69.0%	70.1%	—	68.5%	72.8%	66.7%	—	—
L1 (sparse coding)	72.3%	72.0%	73.4%	74.0%	81.4%	80.6%	83.4%	83.7%	84.7%

MFCC describes the spectral envelope of audio signals in a compact way, it seems to work nicely with *k*means+L2.

On the other hand, using spectrogram as the feature representation for ODL+L1 significantly (p-value<5%) outperforms all the other feature representations under the two-tailed t-test. This result is surprising at the first glance as spectrogram is considered as the most "primitive" representation of music. Yet this result implies that sparse coding works best with a feature representation that captures all the details of the raw signal. Due to its robustness to outliers [52], sparse coding effectively exploits the rich information contained in the spectrogram for signal decomposition while limiting the effect of noises. A similar observation that spectrogram works better than MFCC for sparse representation of music has also been made by Lee *et al.* for exemplar-based multipitch estimation [21].

We also found that it is important to use the log-power spectrogram instead of the raw power spectrogram; using the latter drops the classification accuracy to 78.0%, which is still superior to other features but is significantly worse than the log-power one. Interestingly, this result implies that we can better approximate the spectral magnitude of an input signal by *multiplying* rather than *summing* the spectral magnitude of other signals, as $\log(x) + \log(y) = \log(xy)$.

In addition, we found that the performance of ODL+L1 is less sensitive to the frame size and hop size of the local feature descriptor. We have tested the common settings in the MIR literature [14,16,49], including 512, 1,024 and 2,048 samples for frame size and 1/2, 1/4, and 0 overlapping, and found that the difference between settings is not significant. The performance ranges from 80.1% to 83.4%; using 2,048 samples in a frame with half overlapping performs the best.

Finally, we compared the result of using different frequency bands for ODL+L1. The accuracies of the four settings described in Section 5.1 are 71.2%, 74.3%, 82.6%, and 75.1%, respectively. Using all frequencies (i.e., 0–11k Hz) achieves the best result, whereas limiting the frequency bands only degrades the performance. Dividing frequency ranges is neither effective. Again, sparse coding works better when all available information of the raw signal is exploited.

6.2 Comparison of Encoding Methods

Next, we evaluate the performance of different combinations of codebook generators and codeword encoders using the top two performing feature representations MFCC and spectrogram. The result for GTZAN is shown in Table 2, from which the following observations can be made. First, we see that the sparse coding (L2) consistently outperforms VQ (L1) regardless of the codebook generation methods. The performance difference is 2–3% for MFCC and 7.8–16.7% for spectrogram. The performance difference between L2+spectrogram and L1+spectrogram is significant (p-value<5%) under the two-tailed t-test. Second, when L2 is used as the encoder, there is no major difference between different codebook generators and feature representations.

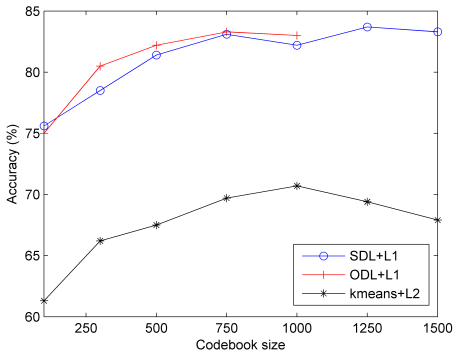


Figure 3: Classification accuracy as we vary the codebook size for different encoding methods.

However, when L1 is used, the performance difference between MFCC and spectrogram is significant, which is in line with our observation in Section 6.1. In addition, dictionary learning methods (ODL, SDL♯, and SDL) generally outperform *k*means and Exemplar. The best result 84.7±4.6% is obtained by SDL+L1+spectrogram. The standard deviation in classification accuracy of the ten folds is around 3–5% for all the considered encoding methods. We do not observe any particular method that leads to better stability.

For the L2 encoder, the best-performing codebook generator is Exemplar+L2+spectrogram, which outperforms the classic *k*means+L2+MFCC [31] by 2.7%. We found that ODL+L2 does not perform well, especially when using spectrogram as the feature representation.

For the L1 encoder, dictionary learning methods generally perform better. Our result shows that SDL is more effective with the non-negativity constraint on α. Although the performance difference between SDL and ODL is not significant (84.7% versus 83.4%), we consider the result remarkable as it outperforms the state-of-the-art accuracy 84.3% obtained by Hamel *et al.* [13] and Panagakis *et al.* [38], who employed much more sophisticated designs such as three-layer deep believe networks or tensor-based feature representation.

As for ISMIR2004Genre, the classification accuracies of ODL, SDL♯, and SDL are 90.4%, 90.12%, and 90.81%, respectively, when using L1 encoder and spectrogram feature representation. These accuracies are significantly better than the existing results 83.2% and 83.5% reported in [38] and [18]. In addition, we note that SDL consistently outperforms ODL for both GTZAN and ISMIR2004Genre.

Fig. 3 shows the result as we vary the size of the codebook with an increasing step of 250. We observe that the performance of SDL+L1 grows linearly as the size of the codebook increases, but reaches a plateau after the codebook size is larger than 750. A similar trend can be observed for ODL+L1. The performance of *k*means+L2 is consistently lower than that of SDL+L1 by around 15%.

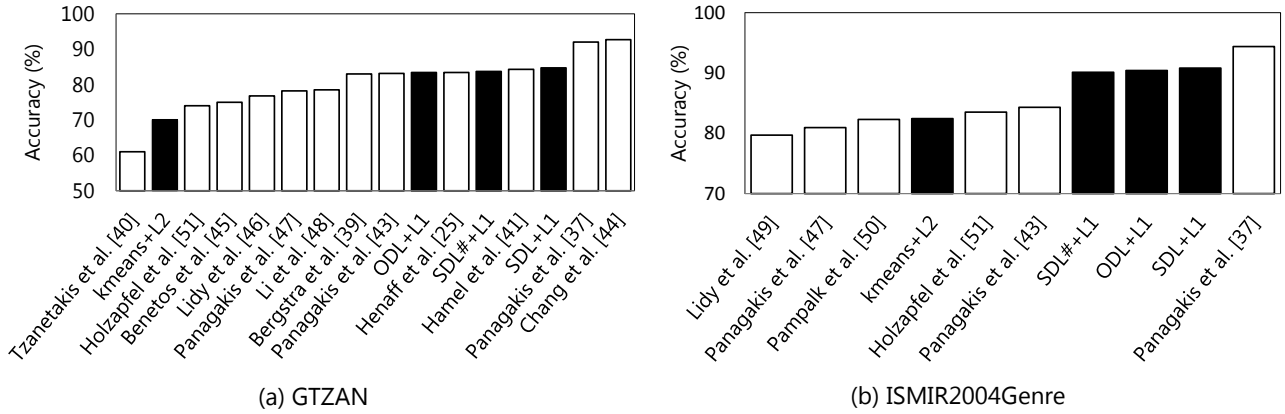


Figure 4: The stat-of-the-art accuracies for (a) GTZAN and (b) ISMIR2004Genre. Black bars are our results.

In terms of efficiency, the computational time for generating a codebook of 2,000 codewords from the short-time features of 900 songs is 46.7 seconds, 90 mins, and 110 mins for $k$means+MFCC, ODL+spectrogram, and SDL+spectrogram, respectively. The average time for encoding a 30-sec clip is 0.569 seconds for L2+MFCC and 15.4 seconds for L1+ spectrogram, both are faster than real-time. As the codebook generation process can be completed in an off-line fashion, the encoding system is considered fairly efficient.

Fig. 4 compares our results with the state-of-the-art. Tzanetakis *et al.* developed one of the first music genre classification systems using three feature sets for representing timbral texture, rhythmic content, and pitch content [49]. Hamel *et al.* employed deep belief network for feature extraction and SVM with radial basis function kernel for classification [13]. Henaff *et al.* used an l_2-based predictive sparse decomposition method to learn a dictionary from a collection of CQT and adopted a linear classifier for classification [16]. Chang *et al.* also computed a sparse representation of music from a large set of short-time and long-time timbral texture features and developed a compressive sampling based classifier [7]. Panagakis *et al.* investigated a bio-inspired third order tensor auditory representation of music signals. They employed non-negative tensor factorization for dimension reduction and shown the proposed method is compatible to the working principle of sparse representation based classification [37, 38]. By far the best results for GTZAN and ISMIR2004Genre are achieved by Chang *et al.* [7] and Panagakis *et al.* [37], respectively. Although the proposed SDL+L1 does not outperform these two prior arts, its performance is on top of the other existing methods.

For future work we would like to exploit more music properties such as the frame dependency of local features [4] and the semantic correlation between music labels [42] to further boost the performance of the presented framework. We are also interested in investigating more dictionary learning algorithms such as deep learning [17] and hierarchical sparse coding [19] and in applying dictionary learning to other MIR tasks such as mood classification [55], music structure analysis [46], and auto-tagging [22, 30, 50].

7. CONCLUSIONS

In this paper, we have presented a dictionary-based framework for summarizing short-time features of music computed over time. We have benchmarked different encoding and codebook construction techniques and demonstrated the superiority of sparsity-enforced dictionary learning to conventional VQ-based or exemplar-based methods. We have also showed that by learning a number of sub-codebooks independently for each class enhances the discriminative power of the encoding system. This supervised dictionary learning algorithm is effective and efficient. Based on the learnt dictionary we obtain 84.7% and 90.8% accuracies for music genre classification on two benchmark datasets GTZAN and ISMIR2004Genre, which are comparable to the state-of-the-art. The presented framework is easy to implement and readily applicable to other multimedia retrieval problems.

8. ACKNOWLEDGMENT

This work was supported by a grant from the National Science Council of Taiwan under NSC 100-2218-E-001-009.

9. REFERENCES

[1] E. Benetos and C. Kotropoulos. A tensor-based approach for automatic music genre classification. In *European Conf. Signal Processing*, 2008.

[2] J. Bergstra and B. Kegl. Aggregate features and adaboost for music classification. In *Machine Learning*, volume 65, pages 473–484, 2006.

[3] J. Bergstra, M. I. Mandel, and D. Eck. Scalable genre and tag prediction with spectral covariance. In *ISMIR*, pages 507–512, 2010.

[4] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Trans. Audio, Speech and Lang. Processing*, 18:538–549, 2010.

[5] T. Bertin-Mahieux and D. Ellis. Large-scale cover song recognition using hashed chroma landmarks. In *IEEE WASPAA*, 2011.

[6] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668–696, 2008.

[7] K. K. Chang, J.-S. R. Jang, and C. S. Iliopoulos. Music genre classification via compressive sampling. In *ISMIR*, pages 387–392, 2010.

[8] S. S. Chen, D. L. Donoho, Michael, and A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Scientific Computing*, 20:33–61, 1998.

[9] B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[10] Z. Fu, G. Lu, K. M. Ting, and D. Zhang. A survey of audio-based music classification and annotation. *IEEE Trans. Multimedia*, 13(99):303–319, 2011.

[11] J. F. Gemmeke, T. Virtanen, and A. Hurmalainen. Exemplar-based sparse representations for noise robust automatic speech recognition. *IEEE Trans. Audio, Speech and Lang. Processing*, 19(7):2067–2080, 2011.

[12] P. Gomez and B. Danuser. Relationships between musical structure and psychophysiological measures of emotion. *Emotion*, 7(2):377–87, 2007.

[13] P. Hamel and D. Eck. Learning features from music audio with deep belief networks. In *ISMIR*, pages 339–344, 2010.

[14] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *ISMIR*, pages 729–734, 2011.

[15] H. Hassanieh, P. Indyk, D. Katabi, and E. Price. Simple and practical algorithm for sparse Fourier transform. In *SODA*, 2012.

[16] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun. Unsupervised learning of sparse features for scalable audio classification. In *ISMIR*, pages 681–686, 2011.

[17] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Comp.*, 18(7):1527–1554, 2006.

[18] A. Holzapfel and Y. Stylianou. Musical genre classification using nonnegative matrix factorization-based features. *IEEE Trans. Audio, Speech and Lang. Processing*, 16(2):424–434, 2008.

[19] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *J. Machine Learning Research*, 2011.

[20] S. Kim, S. Narayanan, and S. Sundaram. Acoustic topic model for audio information retrieval. In *IEEE WASPAA*, pages 37–40, 2009.

[21] C.-T. Lee, Y.-H. Yang, and H. H. Chen. Multipitch estimation of piano music by exemplar-based sparse representation. *IEEE Trans. Multimedia*, 2012. to appear.

[22] M. Levy and M. Sandler. Music information retrieval using social tags and audio. *IEEE Trans. Multimedia*, 11:383–395, 2009.

[23] T. Li, M. Ogihara, and Q. Li. A comparative study on content-based music genre classification. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 282–289, New York, NY, USA, 2003. ACM.

[24] T. Lidy and A. Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification, 2005.

[25] T. Lidy, A. Rauber, A. Pertusa, and J. M. Iñesta. Combining audio and symbolic descriptors for music classification from audio. In *ISMIR*, 2007.

[26] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, 2004.

[27] L. Lu and A. Hanjalic. Audio keywords discovery for text-like audio content analysis and retrieval. *IEEE Trans. Multimedia*, 10(1):74–85, 2008.

[28] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Int. Conf. Machine Learning*, pages 689–696, 2009.

[29] S. Maji, A. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *IEEE CVPR*, pages 1–8, 2008.

[30] M. I. Mandel, D. Eck, and Y. Bengio. Learning tags that vary within a song. In *ISMIR*, pages 399–404, 2010.

[31] B. McFee, L. Barrington, and G. R. G. Lanckriet. Learning content similarity for music recommendation. *CoRR*, abs/1105.2344, 2011.

[32] A. Meng, P. Ahrendt, J. Larsen, and L. K. Hansen. Temporal feature integration for music genre classification. *IEEE Trans. Audio, Speech and Lang. Processing*, 15(5):1654–1664, 2007.

[33] M. Müller, D. P. W. Ellis, A. Klapuri, and G. Richard. Signal processing for music analysis. *J. Sel. Topics Signal Processing*, 5(6):1088–1110, 2011.

[34] E. Pampalk, S. Dixon, and G. Widmer. Exploring music collections by browsing different views. In *ISMIR*, 2003.

[35] E. Pampalk, A. Flexer, and G. Widmer. Improvements of audio-based music similarity and genre classification. In *ISMIR*, pages 628–633, 2005.

[36] I. Panagakis, E. Benetos, and C. Kotropoulos. *Music genre classification: A multilinear approach*, pages 583–588. Citeseer, 2008.

[37] Y. Panagakis, C. Kotropoulos, and G. R. Arce. Music genre classification using locality preserving non-negative tensor factorization and sparse representations. In *ISMIR*, pages 249–254, 2009.

[38] Y. Panagakis, C. Kotropoulos, and G. R. Arce. Non-negative multilinear principal component analysis of auditory temporal modulations for music genre classification. *IEEE Trans. Audio, Speech, and Lang. Processing*, 18(3):576–588, 2010.

[39] J. Paulus, M. Müller, and A. Klapuri. State of the art report: Audio-based music structure analysis. In *ISMIR*, pages 625–636, 2010.

[40] M. D. Plumbley, T. Blumensath, L. Daudet, R. Gribonval, and M. E. Davies. Sparse representations in audio and music: from coding to source separation. *Proceedings of the IEEE*, 98(6):995–1005, 2009.

[41] M. Riley, E. Heinen, and J. Ghosh. A text retrieval approach to content-based audio retrieval. In *ISMIR*, 2008.

[42] C. Sanden and J. Z. Zhang. Enhancing multi-label music genre classification through ensemble techniques. In *SIGIR*, pages 705–714, 2011.

[43] B. Scholköpf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, 2002.

[44] C. Schörkhuber and A. Klapuri. Constant-Q transform toolbox for music processing. In *Sound and Music Computing Conf.*, 2010.

[45] D. Sculley. Web-scale k-means clustering. In *WWW*, pages 1177–1178, 2010.

[46] M.-Y. Su, Y.-H. Yang, Y.-C. Lin, and H.-H. Chen. An integrated approach to music boundary detection. In *ISMIR*, pages 705–710, 2009.

[47] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal Statistical Soc.*, 58:267–288, 1996.

[48] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Towards musical query-by-semantic-description using the CAL500 data set. In *ACM SIGIR*, pages 439–446, 2007.

[49] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. Speech and Audio Processing*, 10(5):293–302, 2002.

[50] 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 *ISMIR*, 2011.

[51] J. Weston, S. Bengio, and P. Hamel. Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval. *J. New Music Res.*, 2012. to appear.

[52] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. Huang, and S. Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, 2010.

[53] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo. Evaluating bag-of-visual-words representations in scene classification. In *MIR*, pages 197–206, 2007.

[54] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE CVPR*, pages 1794–1801, 2009.

[55] Y.-H. Yang and H. H. Chen. *Music Emotion Recognition*. CRC Press, Cambridge, 2011.

[56] B. Zhao, L. Fei-Fei, and E. P. Xing. Online detection of unusual events in videos via dynamic sparse coding. In *IEEE CVPR*, 2011.