**International Journal of Science and Research (IJSR)**
**ISSN (Online): 2319-7064**
**Impact Factor (2012): 3.358**

# Survey of Audio Feature Representation using Encoding Techniques

**Shabnam R. Makandar**[1], **V. L. Kolhe**[2]

[1]Department of Computer Engineering, D. Y. Patil College of Engineering, Akurdi, Pune, Savitribai Phule Pune University, India

**Abstract:** ***Digital music has become prolific in the web in recent decades. Automated recommendation systems are necessary for users to discover music they love and for artists to reach suitable audience. When manual annotations and user preference data is lacking (e.g. for new artists) these systems must rely on content based methods. Besides powerful machine learning tools for classification and retrieval, a key elements for successful recommendation is the audio content representation. Good representations should catch informative musical patterns in the audio signal of songs. These representations should be to the point, to enable efficient (low storage, easy indexing, fast search) management of huge music repositories, and should also be easy and fast to assess, to enable real-time interaction with a user supplying new songs to the system. Before designing new audio features, the usage of traditional local features are explored, while adding a stage of encoding with a pre-computed codebook and a stage of pooling to get compact vectorial representations.***

**Keywords:** Audio content representations, music information retrieval, music recommendation.

## 1. Introduction

Digital music has become more accessible and abundant on the web and large scale systems for recommendation and exploration have become more popular. Since the availability of manual annotations and user preference data is limited (*e.g.* for new, unfamiliar, artists) industrial recommendation systems must incorporate content basedmethods, which interpret the actual audio content of music items and extract meaningful information from it. Content-based systems store information describingthe items, and retrieve items that are similar to those knownto be liked by the user. Items are typically representedby n-dimensional feature vectors.

The most successful approaches to a wide variety of recommendation tasks including not just music, but books, movies, etc.- use collaborative filters (CF). Systems based on collaborative filters exploit the “wisdom of crowds” to infer similarities between items, and recommend new items to users by representing and comparing these items in terms of the people who use them. Within the domain of music information retrieval, recent studies have shown that CF systems consistently outperform alternative methods for playlist generation and semantic annotation. However, collaborative filters suffer from the dreaded “cold start” problem: a new item cannot be recommended until it has been purchased, and it is less likely to be purchased if it is never recommended. Thus, only a tiny fraction of songs may be recommended, making it difficult for users to explore and discover new music.

The cold-start problem has motivated researchers to improve content-based recommendation systems. Content-based systems operate on music representations that are extracted automatically from the audio content, eliminating the need for human feedback and annotation when computing similarity. While this approach naturally extends to any item regardless of popularity, the construction of features and definition of *similarity* in these systems are frequently ad-hoc and not explicitly optimized for the specific task [1].

In the past decade much research was dedicated to constructing content based systems for music information retrieval (MIR) tasks such as music classification (to artist, genre, *etc.*), semantic annotation (auto-tagging) and retrieval (query-by-tag) and music similarity for song-to-song recommendation. The focus was mostly on machine learning algorithms that utilize basic audio features to perform the MIR task.

## 2. Literature Review

### 2.1 Sparse Coding (SC)

M. D. Plumbley et al.[16], has presented their work, which describe an approach to musical audio analysis basedon a search for sparse representations, where any coefficient in such a representation has only a small probability of being far from zero. For music, it is not surprising that a musical audio signal would be generated from a small number of possible notes active at any one time, and hence allow a sparse representation. For example, for a standard pianothere are 88 possible notes that could be played, with each note producing a particular sound at a particular pitch. However, in most piano pieces only a few (e.g. up to 4–6) of the notes are played at any one time, typically limited by the chords (sets of simultaneous notes) desired by the composer, as well as the physical limit on the number of fingers available to the pianist. This leads to the idea that music is sparse, in the sense that at a given time instant most of the available notes are not sounding.

### 2.2 Deep Belief Networks (DBN)

Lee et al.[17] reported the first study that applies DBN to MIR problems. Using the features learnt by DBN outperforms standard acoustic features such as spectrogram and MFCC for both genre classification and singeridentification. DBN is probabilistic, multilayer neuralnetwork that processes information by multiple levels of transformation and abstraction, as different areas of cortex in the mammal brain perform [18]. The idea of DBN is to form a hierarchical signal processing paradigm that mimics how people organize and perceive music information. Many variants of DBN, such as convolutional DBN, conditional DBN, and convolutional neural networks (CNN), have also been utilized for MIR

### 2.3 Bag of Frames (BoF)

J.J. Aucouturier et al [19], has presented their work on a codebook, where 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. 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. 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.

## 3. Song Representation

The encoding-pooling schemes to get a compact representation for each song (or musical piece) are examined. The scheme is comprised of three stages:

**3.1 Short time frame features:** each song is processed to a time series of low-level feature vectors $\in \mathbb{R}^{d\times T}$ , ( $T$ time frames, with a dimensional feature vector from each frame).

**3.2 Encoding:** each feature vector $x_t \in \mathbb{R}^{d}$ is then encoded to $c_t \in \mathbb{R}^{k}$ a code vector, using a pre-calculated dictionary $D \in \mathbb{R}^{d\times k}$, a codebook of “basis vectors” of dimension .We get the encoded song $C \in \mathbb{R}^{k\times T}$ .

**3.3 Pooling:** the coded frame vectors are pooled together to a single compact vector $\in \mathbb{R}^{k}$ .

This approach is known as the bag of features (BoF) approach: where features are collected from different patches of an object (small two-dimensional patches of an image, short time frames of a song, *etc.*) to form a variable-size set of detected features. The pooling stage enables us to have a unified dimension to the representations of all songs, regardless of the songs’ durations. A common way to pool the low-level frame vectors together is to take some statistic of them, typically their mean.

## 4. Musical Features

Audio pre-processing required transforming the audio data into a format suitable for similarity estimation.

### 4.1 Feature Extraction

Feature extraction is the process of computing a compact numerical representation that can be used to characterize a segment of audio. The design of descriptive features for a specific application is the main challenge in building pattern recognition systems. Once the features are extracted standard machine learning techniques which are independent of the specific application area can be used.

#### 4.1.1 Timbral Texture Features
The features used to represent timbral texture are based on standard features proposed for music-speech discrimination and speech recognition. The calculated features are based on the short time Fourier transform (STFT) and are calculated for every short-time frame of sound. More details regarding the STFT algorithm and the Mel-frequency cepstral coefficients (MFCC). The use of MFCC’s to separate music and speech [2].

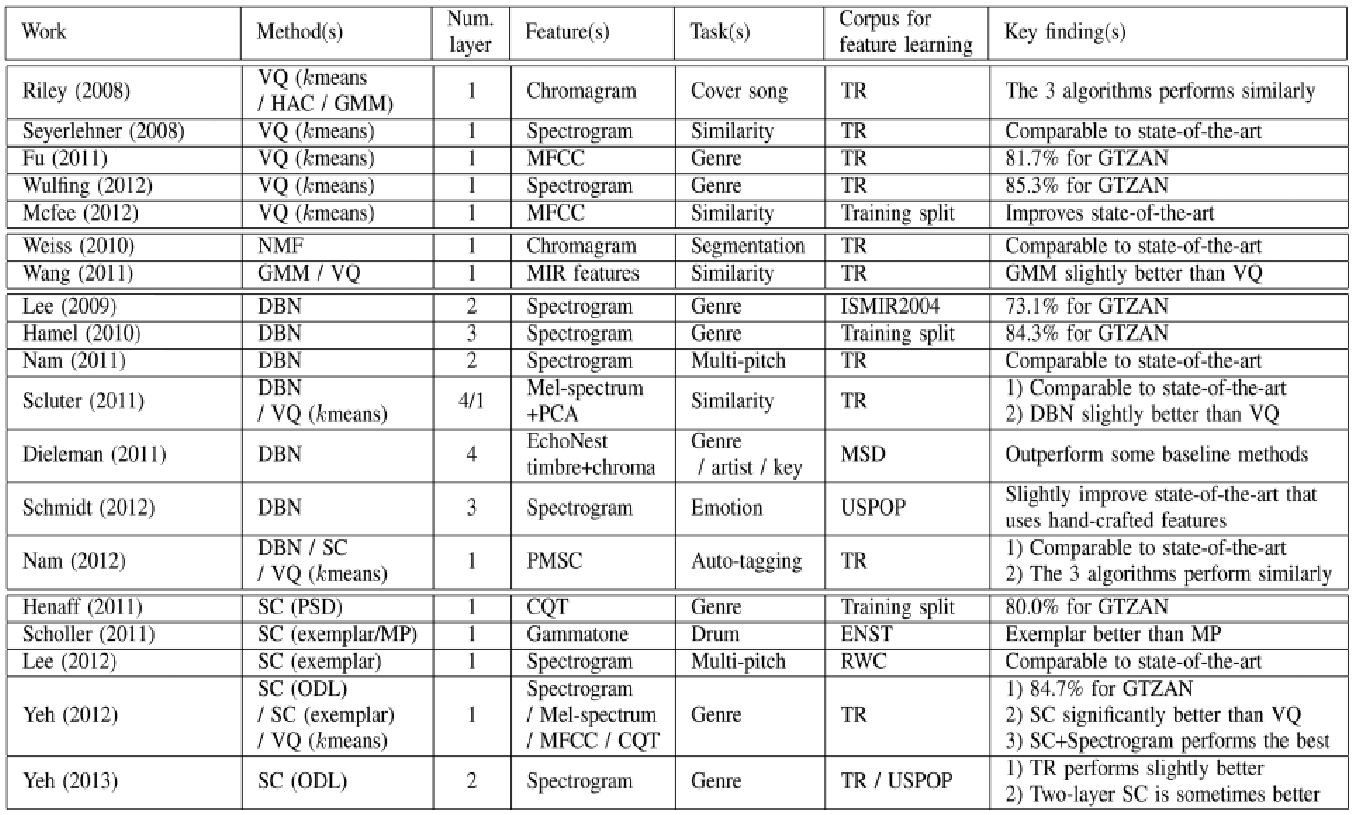
#### 4.1.2 Mel-Frequency Cepstral Coefficients
Mel-frequency cepstral coefficients (MFCC) are perceptually motivated features that are also based on the STFT. After taking the log-amplitude of the magnitude spectrum, the FFT bins are grouped and smoothed according to the perceptually motivated Mel-frequency scaling. Finally, in order to de-correlate the resulting feature vectors a discrete cosine transform is performed [2].

The MFCC vectors are augmented with the following coefficients, which give different descriptions of the distribution of the magnitude spectrum of the audio [5].

- Flux: the difference between consecutive spectra.
- Centroid: the centre of gravity of the spectrum.
- RMS: the Euclidean norm of the spectrum.
- Entropy: the entropy of the spectrum.
- Variance: the estimated sample variance of the spectrum.
- Skew: the estimated sample skew of the spectrum.
- Kurtosis: the estimated sample kurtosis of the spectrum.
- Roll-off: the point on the frequency scale below which 85% of the total amplitude lies.

#### 4.1.3 Timbral Texture Feature Vector
The feature vector for describing timbral texture consists of the following features: means and variances of spectral centroid, roll-off, flux, zero-crossings over the texture window, low energy, and means and variances of the first five MFCC coefficients over the texture window (excluding the coefficient corresponding to the DC component) resulting in a 19-dimensional feature vector [2].

**Table 1**: Comparison of Bag-of-Frmaes based approaches for MIR problems [20], TR indicates Transductive Learning



| Work | Method(s) | Num. layer | Feature(s) | Task(s) | Corpus for feature learning | Key finding(s) |
|---|---|---|---|---|---|---|
| Riley (2008) | VQ (*k*means / HAC / GMM) | 1 | Chromagram | Cover song | TR | The 3 algorithms performs similarly |
| Seyerlehner (2008) | VQ (*k*means) | 1 | Spectrogram | Similarity | TR | Comparable to state-of-the-art |
| Fu (2011) | VQ (*k*means) | 1 | MFCC | Genre | TR | 81.7% for GTZAN |
| Wulfing (2012) | VQ (*k*means) | 1 | Spectrogram | Genre | TR | 85.3% for GTZAN |
| Mcfee (2012) | VQ (*k*means) | 1 | MFCC | Similarity | Training split | Improves state-of-the-art |
| Weiss (2010) | NMF | 1 | Chromagram | Segmentation | TR | Comparable to state-of-the-art |
| Wang (2011) | GMM / VQ | 1 | MIR features | Similarity | TR | GMM slightly better than VQ |
| Lee (2009) | DBN | 2 | Spectrogram | Genre | ISMIR2004 | 73.1% for GTZAN |
| Hamel (2010) | DBN | 3 | Spectrogram | Genre | Training split | 84.3% for GTZAN |
| Nam (2011) | DBN | 2 | Spectrogram | Multi-pitch | TR | Comparable to state-of-the-art |
| Scluter (2011) | 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) | DBN | 4 | EchoNest timbre+chroma | Genre / artist / key | MSD | Outperform some baseline methods |
| Schmidt (2012) | DBN | 3 | Spectrogram | Emotion | USPOP | Slightly improve state-of-the-art that uses hand-crafted features |
| Nam (2012) | 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) | SC (PSD) | 1 | CQT | Genre | Training split | 80.0% for GTZAN |
| Scholler (2011) | SC (exemplar/MP) | 1 | Gammatone | Drum | ENST | Exemplar better than MP |
| Lee (2012) | SC (exemplar) | 1 | Spectrogram | Multi-pitch | RWC | Comparable to state-of-the-art |
| Yeh (2012) | 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) | SC (ODL) | 2 | Spectrogram | Genre | TR / USPOP | 1) TR performs slightly better 2) Two-layer SC is sometimes better |

### 4.2 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. It describes the time-varying energy across different frequency bands in a linear frequency scale of the signal, consider the following other variations,

1. *Mel-spectrogram:* Mel-spectrogram 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.
2. *Sonogram:* It employs techniques such as outer-ear model, Bark-scale critical-bands, and spectral masking to better respect human loudness sensation.
3. *Constant-Q transform:* Constant-Q transform replaces linear frequency scale by a logarithmic one to respect the "octave equivalence" of music perception, i.e., each doubling in frequency corresponds to an equal musical interval [4].
4. *Chroma features:* Chroma features represent of the harmonic content of a short-time window of audio by computing the spectral energy present at frequencies that correspond to each of the 12 notes in a standard chromatic scale [3].

## 5. Temporal Integration

Temporal integration (pooling) is getting a compact representation of a song by generative modelling. In this approach the whole song is described using a parametric structure that models how the song's feature vector time series was generated. Various generative models were used:

### 5.1 Gaussian Mixture Model (GMM)

It is a probabilistic model is one word-level distribution over the audio feature space for each word in our vocabulary. Each distribution is modelled using a multivariate Gaussian mixture model (GMM). The parameters of a word-level GMM are estimated using audio content from a set of training tracks that are positively associated with the word. Using this model, one can infer likely semantic annotations given a novel track and can use a text-based query to rank-order a set of un-annotated tracks [6].

### 5.2 Dynamic Texture Mixture (DTM) model

The dynamic texture (DT) model, a generative time series model that captures longer term time dependencies, for automatic tagging of musical content

Since musical time series often show significant structural changes within a single song and have dynamics that are only locally homogeneous, a single DT would be insufficiently rich to model individual songs and, therefore, the typical musical content associated with semantic tags. To address this at the song-level, Barrington *et al.* propose to model the audio fragments from a *single song* as samples from a dynamic texture mixture (DTM) model, for the task of automatic music *segmentation*. Their results demonstrated that the DTM provides an accurate segmentation of music into homogeneous, perceptually similar segments (corresponding to what a human listener would label as "chorus," "verse," "bridge," etc.) by capturing *temporal* as well as *textural* aspects of the musical signal [7].

### 5.3 Multivariate Autoregressive Model (MAR)

The multivariate auto regressive model handles both temporal and correlations among feature dimensions, which makes it a good candidate for feature integration. A simple autoregressive model was suggested simple refers to considering each feature dimension independently. The MAR model is popular in time-series modelling and prediction being both simple and well understood [8].

### 5.4 Autoregressive Mixture Model (ARM)

An ARM model treats a group of audio fragments as samples from *K* AR models. Specifically, for a given sequence, an assignment variable z ~categorical ($\pi_1, \cdots, \pi_K$) selects one of the *K* AR models, where the $i^{th}$AR model is selected with probability $\pi_i$. The ARM model is a more appropriate modelling choice for an entire song. This is motivated by the observation that a song usually shows significant structural variations within its duration, and hence multiple AR components are necessary to model the heterogeneous sections [9].

### 5.5 Hidden Markov Model (HMM)

Late temporal integration does not try to explicitly extract the feature dynamics. It operates at the classifier level, either by operating a "fusion" of successive primary decisions of the classifier, or by exploiting a classifier that can handle sequences. The usual way of combining several decisions of the classifier is to compute the product of the posteriors for each class, with the implicit assumption that the observations are independent. But because of this assumption, this approach does not capture any information about the temporal evolution of the features. The most popular way to overcome this problem is probably the use of HMM classifiers examples, which handle the sequentiality of the features by fitting a generative model to the features' temporal evolution [13].

### 5.6 Hierarchical Dirichlet Process (HDP)

The Hierarchical Dirichlet Process (HDP) [8] is a model of grouped data, which is more appropriate than the DPMM for modelling songs represented as a collection of MFCCs. Rather than associate each song with a single table in the restaurant, each song is represented as a group of features which sit at a song-specific "local" restaurant. The dishes for this restaurant, however, are drawn from a "global" set of dishes. Thus, each song is represented as a distribution over latent components, but the population of latent components is shared across songs. Similarity between songs can be defined according to the similarity between their corresponding distributions over components. The generative process underlying the HDP can be understood with the Chinese Restaurant Franchise (CRF) [14].

### 5.7 Product Probability Kernel (PPK)

The product probability kernel introduced measures the distance between probability models of the feature vectors. Other divergence based kernels have been suggested, for measuring a similar distance. With the product probability kernel, a closed form solution can be determined for e.g. a mixture of Gaussian, furthermore, the PPK fulfils the requirement for a kernel to be positive semi-definite [8].

## 6. Dictionary Training

For Sparse coding methods the problem of finding the optimal dictionary codewords and code coefficients is a smooth but jointly non-convex problem. The training of the dictionaries (codebooks) is performed with the online learning algorithm.

### 6.1 Codebook Generation

As The following codebook generation methods are considered [4]:

1. **K-means:** generates a codebook by grouping the training data into k clusters according to l2 distance, with each cluster center corresponding to a codeword. Regard k-means as the baseline as it is by far the most common codebook generation method in the literature. 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.
2. **ODL:** The online dictionary learning algorithm. While k-means can be thought as adapting the codebook to the training data for the l2 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. Note that ODL does not use a non-negativity constraint.
3. **SDL:** The proposed supervised dictionary learning algorithm. Hypothetically it outperforms ODL for supervised tasks. As a slight abuse of terminology, use SDL to indicate the version without using non-negativity constraint.
4. **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 and automatic speech recognition. 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.

### 6.2 Codeword Encoding

The following two encoding methods are used [4]:

1. **L2:** based encoding, or vector quantization, is perhaps the most common way for codeword encoding. It encodes a given signal x by solving the following constrained minimization problem,

$$\alpha^* = \arg\min_{||\alpha||_0=1} ||x - D_\alpha||_2, \qquad (1)$$

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

2. **L1:** based encoding obtains a sparse coding α of x by solving Equation.
$$\alpha^* = arg \min_{\alpha} \frac{1}{2}\left|\left|x - D_\alpha\right|\right|_2^2 + \lambda\left|\left|\alpha\right|\right|_1. \qquad (2)$$

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 to achieve L1-based encoding for it has a C-based implementation that is efficient and publicly available.

## 7. Encoding

Encoding of low-level features using pre-calculated codebook was examined for audio and music. This section makes a brief review of the main encoding techniques used in codebook generation for music retrieval.

### 7.1 Encoding with the LASSO

The least absolute shrinkage and selection operator (the LASSO) was suggested as an optimization criterion for linear regression that selects only few of the regression coefficients to have effective magnitude, while the rest of the coefficients are either shrunk or even nullified. The LASSO does that by balancing between the regression error (squared error) and an norm penalty over the regression coefficients, which typically generates sparse coefficients. Usage of the LASSO's regression coefficients as a representation of the input is often referred to as "sparse coding"[11]. The encoding of a feature vector $x_t$ using the LASSO criterion is:

$$c_t = arg \min_{c\in\mathbb{R}^k} \frac{1}{2}\left|\left|x_t - D_c\right|\right|_2^2 + \lambda\left|\left|c\right|\right|_1. \qquad (3)$$

Intuitively it seems that such a sparse linear combination might represent separation of the music signal to meaningful components (*e.g.* separate instruments). However, this is not necessarily the case since the LASSO allows coefficients to be negative and the subtraction of codewords from the linear combination has little physical interpretability when describing how musical sounds are generated. To solve the LASSO optimization problem we use the alternating direction method of multipliers (ADMM) algorithm.

### 7.2 Encoding with Vector Quantization (VQ)

In vector quantization (VQ) a continuous multi-dimensional vector space is partitioned to a discrete finite set of bins, each having its own representative vector. The training of a VQ codebook is essentially a clustering that describes the distribution of vectors in the space. During encoding, each frame's feature vector is quantized to the closest codeword in the codebook, meaning it is encoded as , a sparse binary vector with just a single "on" value, in the index of the codeword that has smallest distance to it (we use Euclidean distance)[12]. It is also possible to use a softer version of VQ, selecting for each feature vector the nearest neighbors among the codewords, creating a code vector $c_t$ with τ "on" values and *k*-τ "off" values:

$$c_t(j) = \frac{1}{\tau}\mathbb{1}\left[D_j \in \tau - nearest\ neighbors\ of\ x_t\right], \qquad (4)$$

$$j \in \{1,2,\ldots,k\}.$$

The hard threshold of selecting just one codeword will result in distorted, noise-sensitive code, while using top quantization will be more robust ([1],[10]).

### 7.3 Encoding with Cosine Similarity (CS)

An alternative to VQ another form of encoding, where each dictionary codeword is being used as a linear filter over the feature vectors: instead of calculating the *distance* between each feature vector and each codeword (as done in VQ), which calculate a *similarity* between them—the (normalized) dot product between the feature vector and the codeword:$\frac{\langle x_t, D_j\rangle}{||x_t||_2}$,.The codewords act as pattern matching filters, where frames with close patterns get higher response.

For each frame, selecting the closest (by Euclidean distance) codeword is equivalent to selecting the codeword with largest CS with the frame. So CS can serve as a softer version of VQ. The L2 normalization of each frame (to get CS instead of unnormalized dot product) is important to avoid having a bias towards frames that have large magnitudes, and can dominate over all other frames in the pooling stage [15].

## 8. Conclusion

In this paper, encoding techniques are explained for the codebook generation for compact representation. Collaborative filters form the basis of state-of-the-art recommendation systems, but cannot directly form recommendations or answer queries for items which have not yet been consumed or rated. By optimizing content-based similarity from a collaborative filter, we provide a simple mechanism for alleviating the cold-start problem and extending music recommendation to novel or less known songs.

Increasing the codebook size results in improved performance for all the encoding methods. The LASSO and CS are inconsistent with regard to the preferred pooling method (mean or max-abs). For all the encoding methods the performance deteriorates when the encoding parameter has too high or too low values. While the LASSO and CS can suffer sharp decrease in performance when adjusting their parameters, VQ is more robust, having smooth and controlled change in performance when adjusting its density parameter τ.

## 9. Acknowledgement

The authors would like to thank the publisher and the researchers for making their resources available. We also thank the college authority for providing the required infrastructure and support. Finally, we would like to extend our heartfelt gratitude to friends and family members.

## References

[1] McFee, L. Barrington, and Lanckriet, “Learning content similarity for music recommendation,” *IEEE Trans. Audio, Speech, Lang.Process.*, vol. 20, no. 8, pp. 2207–2218, Oct. 2012.
[2] Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Trans. Speech Audio Process.*, vol. 10, no. 5, pp. 293–302, Jul. 2002.
[3] L. Barrington,M. Yazdani, D. Turnbull, and G. Lanckriet, “Combining feature kernels for semantic music retrieval,” in *Proc. Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, 2008, pp. 723–728.
[4] Yeh, M. C. , and Y. H. Yang, “Supervised dictionary learning for music genre classification,” in *Proc. ICMR*, 2012.
[5] West, K.; Cox, S., "Incorporating Cultural Representations of Features Into Audio Music Similarity Estimation," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol.18, no.3, pp.625,637, March 2010.
[6] Turnbull, L. Barrington, D. Torres, and G. Lanckriet, “Semantic annotation and retrieval of music and sound effects,” *IEEE Trans. Audio,Speech, Lang. Process.*, vol. 16, no. 2, pp. 467–476, Feb. 2008.
[7] Coviello, A. Chan, and G. Lanckriet, “Time series models for semanticmusic annotation,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 5, pp. 1343–1359, Jul. 2011.
[8] Meng and J. Shawe-Taylor, “An investigation of feature models for music genre classification is using the support vector classifier,” in *Proc.Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, 2005, pp. 604–609.
[9] Coviello, Y. Vaizman, A. B. Chan, and G. Lanckriet, “Multivariate autoregressive mixture models for music autotagging,” in *Proc. 13th Int. Soc. Music Inf. Retrieval Conf. (ISMIR ’12)*, 2012.
[10] R. Lyon, M. Rehn, S. Bengio, T. C.Walters, and G. Chechik, “Sound retrieval and ranking using sparse auditory representations,” *Neural Comput.*, vol. 22, no. 9, pp. 2390–2416, Sep. 2010.
[11] R. Grosse, R. Raina, H. Kwong, and A. Y. Ng, “Shift-invariant sparse coding for audio classification,” in *Proc. Conf. Uncertainty AI*, 2007.
[12] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *J. R.Statist. Soc. Seri. B (Methodol.)*, vol. 58, no. 1, pp. 267–288, 1996.
[13] J. S. Essid and G. Richard, “Temporal integration for audio classification with application to musical instrument classification,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 17, no. 1, pp. 174–186, Jan.2009.
[14] M. Hoffman, D. Blei, and P. Cook, “Content-based musical similarity computation using the hierarchical Dirichlet process,” in *Proc. Int. Soc.Music Inf. Retrieval Conf. (ISMIR)*, 2008, pp. 349–354.
[15] Vaizman, Y.; McFee, B.; Lanckriet, G., "Codebook-Based Audio Feature Representation for Music Information Retrieval," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* , vol.22, no.10, pp.1483,1493, Oct. 2014
[16] 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.
[17] 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.
[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] Li Su; Yeh, C.-C.M.; Jen-Yu Liu; Ju-Chiang Wang; Yi-Hsuan Yang, "A Systematic Evaluation of the Bag-of-Frames Representation for Music Information Retrieval," *Multimedia, IEEE Transactions on* , vol.16, no.5, pp.1188,1200, Aug. 2014.