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 (π_1,· · ·,π_K) selects one of the *K* AR models, where the i^{th}AR model is selected with probability π_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,

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

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.

$$\alpha^* = arg\min_{\alpha} \frac{1}{2}\left||x - D_\alpha|\right|_2^2 + \lambda\left||\alpha|\right|_1. \quad (2)$$

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:

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

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:

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

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

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.