MIREX 2011 AMS - AUDIO SIMILARITY VIA METRIC LEARNING

Brian McFee
Computer Science and Engineering
University of California, San Diego
bmcfee@cs.ucsd.edu

Gert Lanckriet
Electrical and Computer Engineering
University of California, San Diego
gert@ece.ucsd.edu

ABSTRACT

Our submissions (ML1, ML2, ML3) to the Audio Music Similarity (AMS) task are based upon learning an optimal distance metric over vector quantized MFCC histograms. ML1 is optimized to predict similarity derived from a collaborative filter; ML2 is optimized to predict genre similarity; ML3 is an unsupervised baseline which uses a native distance metric. This abstract details the system architecture and parameter settings.

1. INTRODUCTION

Our audio music similarity system (ML1) is motivated by the observation that systems built upon collaborative filters (CF) frequently out-perform competing methods based on audio content or semantic annotations [1, 4, 8]. Because CF methods fail on out-of-sample or long-tail content, we introduced a machine learning framework that optimizes the distance between content-based audio representations in order to predict songs by similar artists [5]. In this framework, the similarity between artists (songs) is determined by the fraction of *users* shared between them (their artists), and not by abstract notions of genre.

For comparison purposes, we also include a submission using a metric trained to predict genre similarity (ML2).

Each song in our framework is summarized by a vector quantization (VQ) histogram, and similarity is determined by Euclidean distance after applying a non-linear kernel transformation and learned linear transformation matrix. To establish a baseline, our third submission (ML3) uses the raw kernel distances between histograms without a learned, optimal transformation.

2. SIMILARITY PIPELINE

The architecture of our similarity system is depicted in Figure 1. In this section, we assume that all parameters have been determined, and focus on the "testing" path, which can be decomposed into three phases: feature extraction, vector quantization, and kernel projection.

2.1 MFCC feature extraction

Given a song's waveform, we first down-sample to 22050Hz and extract the time series of the first 13 Mel frequency cepstral coefficients (MFCC), using half-overlapping 23ms windows [7]. This time series is then augmented with the first and second instantaneous derivatives, resulting in a time series of ΔMFCCs $X \in \mathbb{R}^{39 \times T}$ for a song of T frames. Finally, each ΔMFCC vector X_t is normalized by z-scoring:

$$X_t \mapsto \mathrm{diag}(\sigma)^{-1}(X_t - \mu), \tag{1}$$

where $\mu, \sigma \in \mathbb{R}^{39}$ denote the vectors of coordinate-wise means and standard deviations as estimated on the training set (see Section 3.1).

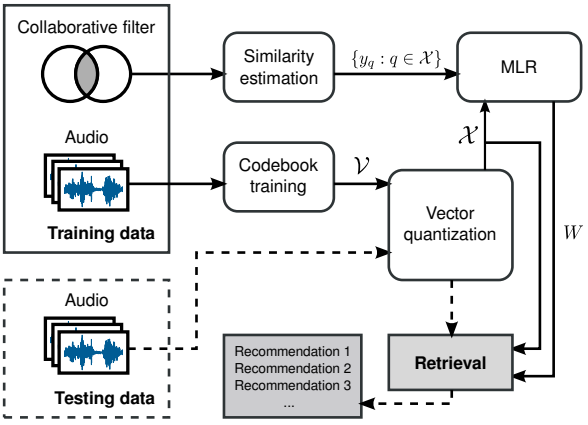

Figure 1. Block diagram of the audio similarity architecture. See [5] for details.

2.2 Vector quantization

After the time series of ΔMFCC vectors has been extracted and normalized, each vector is then quantized to the closest element from a codebook $\mathcal{V}$ of acoustic codewords. This results in a *codeword histogram* representation $h \in [0, 1]^{|\mathcal{V}|}$, where the i^{th} entry records the fraction of times the i^{th} codeword was the quantizer for a frame in X:

$$h_i = \frac{1}{T} \sum_{t=1}^{T} \mathbb{1} \left[v_i = \operatorname*{argmin}_{v \in \mathcal{V}} \|X_t - v\| \right].$$

At this point, each song is represented by a point on the $|\mathcal{V}|$-dimensional probability simplex. In our submission, $\mathcal{V}$ contained 1024 codewords.

2.3 Kernel projection

After computing the codeword histogram h for a song, it is then mapped into a *probability product kernel* (PPK) space [3]. The mapping can be computed explicitly by computing the square root of each coordinate:

$$\widehat{h}_i \leftarrow \sqrt{h_i}.$$

The PPK inner product between two histograms is equivalent to their Bhattacharyya coefficient, and consequently, the Euclidean distance after applying this mapping recovers the same rank-ordering as that derived from Hellinger distance. Moreover, the PPK transformation has been previously observed to provide substantial improvements in accuracy for audio similarity, when combined with metric learning [5].

After applying the PPK transformation, each song's codeword histogram is projected onto the top principal components P which was constructed to capture 95% variance of the training set.

Finally, the compressed codeword histograms are then projected by applying the linear transformation learned by the MLR algorithm (see Section 3.2). The symmetric, positive semi-definite matrix W learned by MLR was factored by spectral decomposition

$$W = V \Lambda V^{\mathsf{T}} \quad \Rightarrow \quad L = \Lambda^{\frac{1}{2}} V^{\mathsf{T}},$$

so that $L^{\mathsf{T}} L = W$. Here, the columns of V contain the eigenvectors of W, and Λ is a diagonal matrix containing the eigenvalues of W (which are non-negative by construction).

The final audio representation is then the composition of L and P with the PPK representation of h:

$$h \mapsto LP\widehat{h}.$$

For a given query song, similar songs are retrieved by rank-ordering the database by increasing Euclidean distance from the query.

3. PARAMETERS

In this section, we describe the data-driven parameters of the system: ΔMFCC statistics μ and σ, the codebook $\mathcal{V}$, PCA matrix P and optimal metric W. For ML1 and ML3, all parameters were estimated from the CAL10K [1] data set [9]. More details can be found in [5]. For ML2, μ, σ and $\mathcal{V}$ were learned from CAL10K, and L and P were learned from the GTZAN genre data set [10].

3.1 Codebook training

The ML1 distance metric is optimized to predict similarities extracted from a sample of Last.fm [2] collaborative filter data [2]. In the first step of training, we partitioned the artists of CAL10K into the *codebook set* and the *experiment set* (all other songs). Artists in the experiment set have at least 100 users that scrobbled 10 or more times; all other artists are grouped into the codebook set.

The codebook set consists of 2646 unique artists and 5513 songs. From each artist, we randomly selected one song, and extracted the ΔMFCC time series from a 5-second clip (431 frames). These clips were aggregated across all codebook artists to form a bag of approximately 1.1 million samples. From this collection, we estimated the mean μ and coordinate-wise standard deviation σ.

Each of the 1.1 million samples was then normalized according to (1), and clustered via online k-means to yield a codebook $\mathcal{V}$ of 1024 cluster centroids. [3]

3.2 ML1: Collaborative filter similarity

To learn the optimal distance metric W from collaborative filter data, we apply the metric learning to rank (MLR) algorithm [6] [4] as follows. The experiment set consisted of 5319 songs by 2015 artists. The artists were then randomly partitioned into training (40%), validation (30%), and test sets (30%), along with their constituent songs.

We then performed principal components analysis on the PPK codeword histograms of training set songs to obtain the matrix P. For the training set used here, 255 components sufficed to capture 95% of variance.

For each training/validation/test artist, the 10 most similar training set artists were found by computing the Jaccard index between user populations in the collaborative filter sample. Songs by the 10 most similar training artists are denoted as *relevant* during MLR training, and all other songs are denoted as irrelevant. To train the metric W, we performed a parameter sweep over the slack trade-off $C \in \{10^{-2}, \ldots, 10^{9}\}$ and ranking loss $\Delta \in \{\text{AUC}, \text{MRR}, \text{NDCG@10}\}$. The W which achieved highest performance on the validation was then applied to the test set, and included in ML1.

3.3 ML2: Genre similarity

Our second submission uses a metric W trained to predict genre similarity. To learn the metric, we used the GTZAN genre set of 1000 tracks in 10 genres [10]. Using the μ, σ and $\mathcal{V}$ described in Section 3.1, each track was processed to yield 1024-dimensional PPK codeword histograms. This set was then projected onto its principal components to yield 317-dimensional compressed representations (again, capturing 95% variance).

Each genre was then partitioned 50/50 into training and validation sets. Following the procedure described in Section 3.2, we learned a metric W by validating over C and Δ

[1] Previously known as SWAT10k

[2] `http://last.fm`

[3] The size of the codebook was varied from 256 to 2048; 1024 yielded the best performance-accuracy tradeoff.

[4] Our MATLAB implementation is freely available at `http://www-cse.ucsd.edu/~bmcfee/code/mlr/`.

and selecting the W which achieved highest AUC on the validation set. Here, relevance was determined by genre agreement instead of CF similarity. The best-performing W was included in ML2.

3.4 ML3: Unsupervised baseline

The ML3 submission is identical to ML1, except that the MLR step is skipped by setting L to the identity matrix. This effectively computes a native distance between low-dimensional projections of PPK codeword histogram representations.

4. REFERENCES

[1] Luke Barrington, Reid Oda, and G.R.G. Lanckriet. Smarter than genius? Human evaluation of music recommender systems. In *ISMIR*, 2009.

[2] O. Celma. *Music Recommendation and Discovery in the Long Tail*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2008.

[3] Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *JMLR*, 5:819–844, Dec 2004.

[4] Joon Hee Kim, Brian Tomasik, and Douglas Turnbull. Using artist similarity to propagate semantic information. In *Proceedings of the 10th International Conference on Music Information Retrieval*, 2009.

[5] B. McFee, L. Barrington, and G.R.G. Lanckriet. Learning content similarity for music recommendation, 2011. http://arxiv.org/1105.2344.

[6] Brian McFee and G.R.G. Lanckriet. Metric learning to rank. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th annual International Conference on Machine Learning (ICML)*, pages 775–782, Haifa, Israel, June 2010.

[7] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[8] M. Slaney and W. White. Similarity based on rating data. In *ISMIR*, pages 479–484, 2007.

[9] D. Tingle, Y. Kim, and D. Turnbull. Exploring automatic music annotation with "acoustically-objective" tags. In *IEEE International Conference on Multimedia Information Retrieval (MIR)*, 2010.

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