SUMMARIZING POPULAR MUSIC VIA STRUCTURAL SIMILARITY ANALYSIS

Matthew Cooper and Jonathan Foote

FX Palo Alto Laboratory
3400 Hillview Ave., Bldg. 4
Palo Alto, CA 94304 USA
[cooper, foote]@fxpal.com

ABSTRACT

We present a framework for summarizing digital media based on structural analysis. Though these methods are applicable to general media, we concentrate here on characterizing repetitive structure in popular music. In the first step, a similarity matrix is calculated from inter-frame spectral similarity. Segment boundaries, such as verse-chorus transitions, are found by correlating a kernel along the diagonal of the matrix. Once segmented, spectral statistics of each segment are computed. In the second step, segments are clustered based on the pairwise similarity of their statistics, using a matrix decomposition approach. Finally, the audio is summarized by combining segments representing the clusters most frequently repeated throughout the piece. We present results on a small corpus showing more than 90% correct detection of verse and chorus segments.

1. INTRODUCTION

Digital music has recently become explosively popular. Many people routinely amass substantial collections of digital audio files. In a recent Ipsos-Reid survey, more than half of consumers aged 25–34 have downloaded MP3 files onto home computers, storing on average more than 700 files [1]. Popular hand-held music players can now store 30 GB of compressed music, or several thousand files. Locating and browsing thousands of tracks is a considerable data management problem, and requires new technologies and tools to support browsing and searching.

Here we present a novel approach to automatically summarize a music track by its most-repeated segments. In popular music, these will be the most memorable parts of the song, typically the verse and chorus. Though there are many variations, a typical song starts with an introductory section, followed by a verse and chorus. These are usually repeated, then followed by a "bridge" or "middle eight [bars]," and often the verse and chorus to end. By representing a longer song by the characteristic verse and chorus, we can enable rapid audio browsing through many files. Automatically inferred structure also enables rapid browsing within an audio file by advancing to the next segment, or to the next segment that is new to the listener.

This paper presents analytic methods to find repetitive structure in digital audio files. Our approach starts with a similarity matrix [2, 3] that captures all possible pairwise similarity measures between time windows in the source audio. A key advantage here is that the data is effectively used to model itself. Throughout, the algorithms presented are unsupervised and need minimal assumptions regarding the source audio. The framework is extremely general, and can be used to analyze any ordered media such as video or text as well as audio.

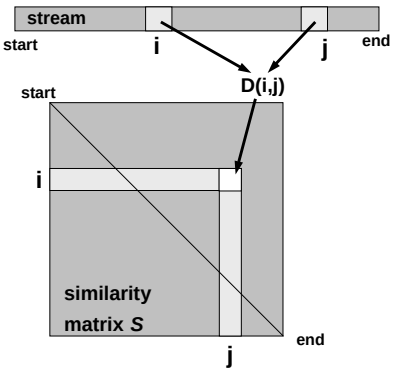

Figure 1: Embedding an audio stream into a two-dimensional similarity matrix.

In this paper, Section 2 reviews related work on summarization and structural analysis. Section 3 introduces similarity analysis and details how similarity matrices are constructed from audio. We also review audio segmentation via kernel correlation. Section 4 describes how audio segments are statistically clustered. We present experimental results showing that segments can be reliably located and classified. Section 5 describes how song structure can be used to automatically construct a representative summary excerpt or "audio thumbnail."

2. RELATED WORK

2.1. Music Summarization

Chu and Logan present methods for music summarization in [5]. The first method clusters uniformly spaced segments. After the audio is parameterized into Mel frequency cepstral coefficients (MFCCs), segments are clustered by thresholding a relative entropy distance measure between them. The longest component of the most frequent cluster is returned as the summary. Key differences with the work in this paper are that our approach uses variable length segments to perform our clustering. As a result, we do not weigh segments' importance by their lengths, but rather by their frequency of repetition. Also, our clustering is based on a matrix decomposition that does not require the use of a threshold. In the second method, they apply a hidden Markov model to jointly segment and cluster the data, based on manually seg-

mented training data. In contrast, our technique uses the digital audio to model itself for both segmentation and clustering. Tzanetakis and Cook [6] discuss "audio thumbnailing" using a segmentation based method in which short segments near segmentation boundaries are concatenated. This is similar to "time-based compression" of speech [7]. In contrast, we use complete segments for summaries, and we do not alter playback speed. Previous work by the authors has also used similarity matrices for excerpting, without an explicit segmentation step [8]. The present method results in a structural characterization, and is far more likely to start or end the summary excerpts on actual segment boundaries. We have also presented an earlier version of this approach, however with less complete validation [4].

2.2. Media Segmentation, Clustering, & Similarity Analysis

Our clustering approach is inspired by methods developed for segmenting still images [9]. Using color, texture, or spatial similarity measures, a similarity matrix is computed between pixel pairs. This similarity matrix is then factorized into eigenvectors and eigenvalues. Ideally, the foreground and background pixels exhibit within-class similarity and between-class dissimilarity. Thus thresholding the eigenvector corresponding to the largest eigenvalue can classify the pixels into foreground and background. In contrast, we employ a related technique to cluster time-ordered data. Gong and Liu have presented an SVD based method for video summarization [10], by factorizing a rectangular time-feature matrix, rather than a square similarity matrix. Cutler and Davis use affinity matrices to analyze periodic motion using a correlation-based method [11].

3. SIMILARITY ANALYSIS

3.1. Constructing the similarity matrix

Similarity analysis is a non-parametric technique for studying the global structure of time-ordered streams. First, we calculate 80-bin spectrograms from the short time Fourier transform (STFT) of 0.05 second non-overlapping frames in the source audio. Each frame is Hamming-windowed, and the logarithm of the magnitude of the FFT is binned into an 80-dimensional vector. We have also experimented with MFCCs and subspace representations computed using principal components analysis of spectral data. Here, the parameterization is optimized for analysis, rather than for compression, transmission, or reconstruction. The sole requirement is that similar source audio samples produce similar features.

The similarity between all pairwise combinations of spectral vectors is quantitatively measured. Represent the B-dimensional spectral data computed for N windows of a digital audio file by the vectors $\{v_i : i = 1, \cdots, N\} \subset I\!R^B$. Using the cosine similarity measure, we embed the resulting similarity data in a square matrix, $\mathbf{S}$ as illustrated in Figure 1. The elements of $\mathbf{S}$ are

$$\mathbf{S}(i,j) = d_{cos}(v_i, v_j) = \frac{< v_i, v_j >}{|v_i||v_j|} \quad . \tag{1}$$

The time axis runs on the horizontal (left to right) and vertical (top to bottom) axes of $\mathbf{S}$ and along its main diagonal, where (self) similarity is maximal. The top panel of Figure 2 shows a similarity matrix computed from the song "The Magical Mystery Tour" by The Beatles using the cosine distance measure and low frequency spectrograms. Pixels are colored brighter with increasing similarity, so that segments of similar audio samples appear as bright squares along the main diagonal. Brighter rectangular regions off the main diagonal indicate similarity between segments.

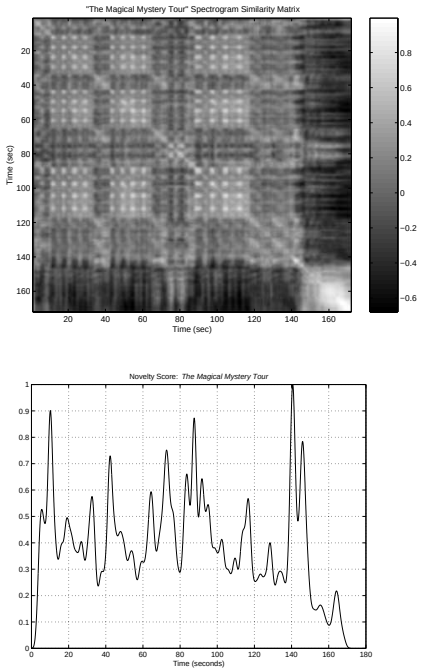

Figure 2: Top: The similarity matrix computed from the song "The Magical Mystery Tour " by The Beatles. Bottom: the time-indexed novelty score produced by correlating the checkerboard kernel along the main diagonal of the similarity matrix.

3.2. Audio Segmentation

The structure of $\mathbf{S}$ can be analyzed to find segment boundaries [12]. Generally, the boundary between two coherent audio segments produces a checkerboard pattern. The two segments will exhibit high within-segment (self) similarity, producing adjacent square regions of high similarity along the main diagonal of $\mathbf{S}$. The two segments will also produce rectangular regions of low between-segment (cross) similarity off the main diagonal. The boundary is the crux of this checkerboard pattern.

To identify these patterns in $\mathbf{S}$, we take a matched-filter approach. A Gaussian-tapered "checkerboard" kernel is correlated along the main diagonal of the similarity matrix. Because peaks in the correlation indicate locally novel audio, we refer to the correlation as a novelty score. Figure 2 shows the similarity matrix computed for "The Magical Mystery Tour" and the corresponding novelty score. Large peaks are detected in the time-indexed correlation and labeled as segment boundaries. For segmentation, we need only calculate a diagonal strip of the similarity matrix with the width of the checkerboard kernel. Throughout, we use a 256×256 kernel.

4. STATISTICAL SEGMENT CLUSTERING

Above, we compute a partial time-indexed similarity matrix to detect audio segment boundaries. In the second step, we use similarity analysis to efficiently cluster the detected segments. This process both locates repeated segments separated in time, and corrects over-segmentation errors. Given segment boundaries, we can easily calculate a full similarity matrix of substantially lower dimension, indexed by segment instead of time. To estimate the similarity between variable length segments, we use a statistical measure.

We assume only that the audio or music exhibits instances of similar segments, possibly separated by other segments. For example, a common popular song structure is **ABABCAB**, where **A** is a verse segment, **B** is the chorus, and **C** is the bridge. We aim to group the segments of such a song into three clusters corresponding to the three different parts. Once this is done, the song could be summarized by concatenating excerpt segments representing each cluster. In this example, the sequence **ABC** is a significantly shorter summary containing essentially all the information in the song.

To cluster the segments, we factor the segment similarity matrix to find repeated or substantially similar groups of segments. The Singular Value Decomposition (SVD) is a natural way to do this. Clustering via factorized similarity matrices is a technique originally developed for still image segmentation; for a survey see [9]. This work suggests that we simply apply the SVD to the full sample-indexed similarity matrix to detect clusters of similar samples, like those visible in Figure 2. This is computationally intensive, however; a three minute song requires computing and factoring a 3600×3600 similarity matrix. This motivates our approach of segmentation followed by segment-level clustering. In this case, the SVD is computed for a similarity matrix whose dimension is on the order of 10×10. This results in an computational savings of several orders of magnitude.

We start with a set of segments $\{p_1, \cdots, p_P\}$ of variable lengths as found above. Each segment is determined by a start and end time. We compute the mean vector, μ_i, and covariance matrix, Σ_i, from the spectral data of each segment, p_i. Inter-segment similarity is calculated using the Kullback-Leibler (KL) distance [13] between these statistics for each pair of segments. Denote the KL distance between the two B-dimensional normal densities $\mathbb{G}(\mu_i, \Sigma_i)$ and $\mathbb{G}(\mu_j, \Sigma_j)$ as $d_{KL}(\mathbb{G}(\mu_i, \Sigma_i) \| \mathbb{G}(\mu_j, \Sigma_j))$. The similarity between segments p_i and p_i is calculated as

$$\begin{aligned} d_{seg}(p_i, p_j) = \exp(& - d_{KL}(\mathbb{G}(\mu_i, \Sigma_i) \| \mathbb{G}(\mu_j, \Sigma_j)) \\ & - d_{KL}(\mathbb{G}(\mu_j, \Sigma_j) \| \mathbb{G}(\mu_i, \Sigma_i))) \ . \end{aligned} \tag{2}$$

where $d_{seg}(\cdot, \cdot) \in (0, 1]$ and is symmetric. These properties are desirable for the clustering technique detailed below.

For clustering, the KL similarity between each pair of segments is embedded in a *segment-indexed* similarity matrix, $\mathbf{S}_S$:

$$\mathbf{S}_S(i, j) = d_{seg}(p_i, p_j) \quad i, j = 1, \cdots, P \ .$$

The top panel of Figure 3 shows the segment similarity matrix $\mathbf{S}_S$ for the song "The Magical Mystery Tour." We then compute the SVD of $\mathbf{S}_S$:

$$\mathbf{S}_S = \mathbf{U} \mathbf{\Lambda} \mathbf{V}^t \ . \tag{3}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices and $\mathbf{\Lambda}$ is a diagonal matrix whose diagonal elements are the singular values of $\mathbf{S}_S$: $\mathbf{\Lambda}_{ii} = \lambda_i$.

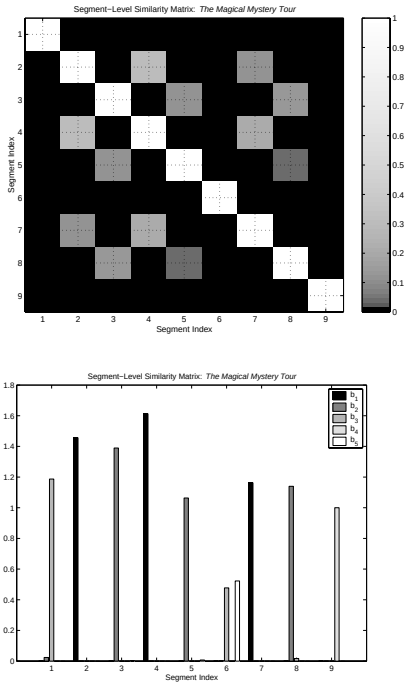

Figure 3: Top: Segment-indexed similarity matrix $\mathbf{S}_S$ computed for "The Magical Mystery Tour" showing repeated segments. Bottom: The bar graph shows the vectors $b_1, \cdots, b_5$ computed according to (6). The maximal vector is used to assign each segment to a segment cluster using Algorithm 1.

We use the singular values to estimate the rank of $\mathbf{S}_S$, denoted K, as the number of singular values greater than one.

The SVD provides the optimal rank-K approximation to $\mathbf{S}_S$:

$$\mathbf{S}_S(i, j) = \sum_{k=1}^{K} \lambda_k \mathbf{U}(i, k) \mathbf{V}(j, k) \tag{4}$$

$$= \sum_{k=1}^{K} \mathbf{B}_k(i, j) \tag{5}$$

The terms in this sum are ordered by decreasing singular value, and hence, by the amount of structure in $\mathbf{S}_S$ for which they account. The rank estimation ensures that we account for the essential structure of the stream, while excluding the fine or unrepeated structures.

To cluster the segments, we sum the rows of the terms in (5):

$$b_k(j) = \sum_{i=1}^{P} \mathbf{B}_k(i, j) \ , \quad j = 1, \cdots, P \ . \tag{6}$$

We evaluate b_k for $k = 1, \cdots, K$. The values of $b_k(j)$ indicate the similarity of segment j to (all) the segments in the k^{th} segment cluster. The maximal vector indicates the cluster to which each segment in the song is assigned. Specifically, we assign the i^{th} segment to cluster k^* such that

$$k^* = \underset{k=1, \cdots, K}{\text{ArgMax}} \, b_k(i) \ . \tag{7}$$

Table 1: Breakdown of experimental results. "V" and "C" columns denote verse and chorus occurrences, respectively.

Song		Manual		Automatic	
Title	Artist	V	C	V	C
Wild Honey	U2	3	3	3	3
Mystery Tour	The Beatles	3	3	3	3
Tahitian Moon	Perry Farell	3	3	3	3
Lucy in the Sky	The Beatles	4	3	3	3
The Zephyr Song	Chili Peppers	3	3	2	3
Hash Pipe	Weezer	2	3	2	3
Optimistic	Radiohead	2	3	2	3

The bottom panel of Figure 3 shows $b_1, \cdots, b_5$ for "The Magical Mystery Tour". In our example, the clusters corresponding to b_1 and b_2 are the verse and chorus clusters, respectively, each including three segments. The introduction of segment 1 is assigned to cluster 3. The bridge segment (segment 6) and coda (segment 9) are assigned to clusters 5 and 4, respectively.

5. STRUCTURAL SUMMARIZATION AND SEGMENTATION RESULTS

In this Section, we explore ways to construct musical summaries given the song structure deduced as above. Intuition suggests that the most-repeated segments would serve as a good summary of the song. Segments in clusters with the largest singular values are most likely to be repeated, so we choose the two with largest singular values. For each of these, we contribute the segment with the maximal value in the corresponding cluster indicator b_i of (6) to the summary. For cluster selection, we have also experimented with the clusters with maximal off-diagonal elements, indicating the segments that are most faithfully repeated in the song.

This is only one of many possible ways to construct audio summaries from the inferred song structure. We could delete all repeated segments, thus ensuring that all the information in the song is included without redundancy. This is analogous to including one segment for each cluster. We could select a subset of these segments to satisfy temporal constraints, such as a maximum summary length. We can also integrate knowledge of the ordering of the segments and clusters, application-specific constraints, or user preferences into the summarization process.

We also use (7) and a heuristic that predicts that the cluster with the first occurring segment is the verse cluster, and the second is the chorus cluster. Applied to a test set of seven popular songs, this approach correctly segmented and labeled over 90% of the verse and chorus segments ($39/43 = 90.7\%$ recall rate within the songs), with no mislabelling of non-verse or non-chorus segments (100 % precision). For brevity, we only list the songs and results in Table 1. Classification was compared with manually-labeled ground truth derived from the automatically detected segments. Space does not permit a fuller discussion, but readers are invited to examine more detailed results, including audio excerpts, on our web site [14].

6. CONCLUSION

We have presented a framework for determining temporal structure from self-similarity. This approach makes minimal assumptions regarding the content or structure of the source. Given an appropriate parameterization and distance measures, these methods could analyze other media types such as video. In future work, we plan to use structural information for music information retrieval and genre classification. Shorter summaries that contain the gist of longer works can usefully serve as "retrieval proxies," where computationally-intensive indexing can be performed on the summary rather than the longer work. Hopefully this will result in more rapid indexing time at no loss in retrieval recall. This application also provides opportunities for objectively measuring summarization success. A particular application of this work is to enable rapid browsing of large music collections, for example on a consumer's handheld MP3 player. By quickly skipping ahead to the next musical section, or playing only the chorus of a song in a "music scan" mode, we expect that these summaries will help users to quickly and easily locate desired music.

7. REFERENCES

[1] Ipsos-Reid, "Digital Music Behavior Continues to Evolve," press release, January 31, 2002. http://www.ipsos-reid.com/media/dsp_displaypr.prnt.cfm?ID_to_view=1414

[2] J. Eckman, et al. "Recurrence Plots of Dynamical Systems," *Europhys. Lett.* **4**, 973, 1987.

[3] K. Church and J. Helfman. "Dotplot: A Program for exploring Self-Similarity in Millions of Lines of Text and Code," *J. American Statistical Assoc.*, **2**(2):153–174, 1993.

[4] J. Foote and M. Cooper. "Media Segmentation using Self-Similarity Decomposition," *Proc. SPIE Storage and Retrieval for Multimedia Databases*, Vol. 5021, pp. 167-75, 2003

[5] S. Chu and B. Logan. "Music Summary Using Key Phrases," *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2000.

[6] G. Tzanetakis and P. Cook. "Audio Information Retrieval (AIR) Tools," *Proc. International Symposium on Music Information Retrieval*, 2000.

[7] L. Stifelman, B. Arons, and C. Schmandt. "The Audio Notebook: Paper and Pen Interaction with Structured Speech," *Proc. ACM CHI* **3**(1):182–189, 2001.

[8] M. Cooper and J. Foote. "Automatic Music Summarization via Similarity Analysis," *Proc. International Symposium on Music Information Retrieval*, pp. 81–5, 2002.

[9] Y. Weiss. "Segmentation Using Eigenvectors: A Unifying View," *Proc. IEEE Intl. Conf. on Computer Vision*, pp. 975–982, 1999.

[10] Y. Gong and X. Liu. "Video Summarization Using Singular Value Decomposition," *Proc. IEEE Intl. Conf. on Computer Vision & Pattern Recognition*, 2000.

[11] R. Cutler and L. Davis. "Robust real-time periodic motion detection, analysis, and applications," *IEEE Trans. Pattern Analysis and Machine Intelligence*. **22**(8):781–796 , 2000.

[12] J. Foote. "Automatic Audio Segmentation using a Measure of Audio Novelty," *Proc. IEEE Intl. Conf. on Multimedia and Expo* **1**:452–455, 2000.

[13] T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

[14] http://www.fxpal.com/media/waspaa03.html