MOBILE MUSIC MODELING, ANALYSIS AND RECOGNITION

Pavel Golik[1*], *Boulos Harb*[2], *Ananya Misra*[2], *Michael Riley*[2], *Alex Rudnick*[3*], *Eugene Weinstein*[2]

[1]Human Language Technology and Pattern Recognition, Computer Science Department,
RWTH Aachen University, 52056 Aachen, Germany
[2]Google Inc., New York, NY 10011, USA
[3]School of Informatics and Computing, Indiana University, Bloomington, IN 47405, USA
golik@cs.rwth-aachen.de, alexr@cs.indiana.edu,
{harb,amisra,riley,weinstein}@google.com

ABSTRACT

We present an analysis of music modeling and recognition techniques in the context of mobile music matching, substantially improving on the techniques presented in [1]. We accomplish this by adapting the features specifically to this task, and by introducing new modeling techniques that enable using a corpus of noisy and channel-distorted data to improve mobile music recognition quality. We report the results of an extensive empirical investigation of the system's robustness under realistic channel effects and distortions. We show an improvement of recognition accuracy by explicit duration modeling of music phonemes and by integrating the expected noise environment into the training process. Finally, we propose the use of frame-to-phoneme alignment for high-level structure analysis of polyphonic music.

Index Terms— Music, modeling, music information retrieval, signal analysis.

1. INTRODUCTION

We explore the problem of identifying musical snippets captured from the surrounding environment using a statistical modeling approach based on automatic speech recognition (ASR). However, unlike natural language, which can be decomposed into *a priori* determined words and phonemes, music in general does not have a universal symbolic transcription that can be similarly used to train a recognition system with annotated corpora.

A variant of the problem was first explored in [2] wherein the snippet is a CD-quality excerpt of the song. In contrast to the ASR approach, most existing music recognition applications make use of Locality Sensitive Hashing [3], creating a fingerprint table of known songs that is queried at run-time for a sequence of feature vectors extracted from the snippet [4, 5]. The acoustic features used are usually designed to be sparse and rely on points of interest such as peaks in spectrograms. This approach has shown robustness to various types of distortions [4], but the rate of false positives can still be high at low SNR necessitating post-processing heuristics. A modeling approach, however, enables noise and channel robustness by allowing these distortions to be modeled, both through the statistical properties of Gaussian Mixture Models (GMM) and by explicitly training on noisy data in addition to or instead of clean data. It also makes it possible to represent the song audio in terms of a low-dimensional alphabet of symbolic units, which provides novel opportunities for music structure analysis.

We extend on the techniques of [2] to allow recognition of snippets captured through a mobile phone's microphone, and we report on the results of an initial study of music analysis using music phoneme acoustic models. In the next section we describe changes to the front end (Section 2.1), acoustic modeling improvements (Section 2.2), and the introduction of duration modeling (Section 2.3). Section 3 presents the use of frame-to-phoneme alignment for high-level audio structure analysis. We present our experimental results in Section 4, and we conclude in Section 5.

2. MODELING

The use of weighted finite-state transducers (WFSTs) for music recognition was introduced in [2, 1]. In this approach, the acoustic model is initialized by unsupervised clustering of the features to form a set of music phonemes. At each iteration of the model training that follows, the acoustic model is refined by alternatively estimating the parameters of the GMMs for each music phoneme and re-transcribing the training data with the model just learned. Unlike in speech recognition, there is no ground truth transcription for the audio. In this process, the transcription used in the acoustic model training is estimated at the same time as the parameters of the GMM. The final WFST for recognition maps subsequences of the final transcription for each song to the song identifiers. The algorithm for constructing this WFST uses weighted determinization and minimization, which preserves the total weight of a path corresponding to a given song. We refer the reader to [1] for details.

The above system achieved 99.9% identification accuracy over test snippets cut from clean recordings, and in [6, 1] we showed that the system was robust to synthetic distortions. Nevertheless, we observed a significant degradation in accuracy when the test recordings were recorded with mobile phones. These recordings are marked with substantial quality degradation of the test audio, a significant spectral tilt introduced by the mobile phone microphone, as well as noise and channel characteristics introduced by recording in a real-world environment. In all the work presented on speech model-based music recognition thus far, clean reference song audio is used to train the acoustic models. However, a model-based approach has the significant advantage of allowing noisy data to be used during training. In addition, the mismatch between the training and test conditions can

* Work was performed at Google. Names are in alphabetical order.

be mitigated by using sound features that are less sensitive to changes in the transmission channel. In this section, we describe our use of these improvements to produce a mobile music recognition system of drastically improved accuracy.

2.1. Acoustic Front-end

Mel-frequency cepstral coefficients (MFCCs) have been broadly used successfully in music processing, e.g., [7]. Typically, the features used are a verbatim copy of those employed in speech recognition. However, it is possible to improve feature quality by making adjustments to the acoustic front-end motivated by the differences between music and speech audio.

As a baseline, we have used a standard speech recognition front end computing perceptual linear predictive (PLP) features [8]. Thirteen cepstral coefficients are computed every 10 ms over 25 ms windows, and the energy coefficient C0 is discarded, which decreases the sensitivity to changes in volume. Even though the training examples were sampled at 16 kHz, we restrict the feature extraction to a range of 125 to 3800 Hz, since we expect this range to be sufficient for song identification. The resulting 36-dimensional feature vector consists of 12 coefficients along with their first and second temporal derivatives.

A speech sound may be accurately identified with a fairly coarse representation of the signal. In contrast, polyphonic music recordings are marked with the presence of more fine-scale acoustic phenomena which necessitate a higher spectral resolution. To accomplish this, we increased the window size from 25 ms to 500 ms and reduced the frequency range to 200 to 2000 Hz. The disadvantage of such a change is that the resulting blurring along the time axis will potentially reduce the capacity of the features to properly represent information about very short sounds. However, overall we have found that this has improved recognition accuracy.

While PLP features are motivated by human perception of speech [9], music has vastly different spectral and temporal characteristics. Thus, conventional MFCC features can be expected to be a more suitable descriptor of the underlying short-time spectrum, since they require fewer assumptions about perception. One of the main assumptions specific to MFCC is the use of the mel-scale, which exploits the fact that the spectral resolution of human hearing decreases exponentially in frequency.

For representing music sounds, we propose a natural modification of the mel-scale that maintains this exponential nature but allows the filterbank channels to be centered at the frequencies of the notes of the equal-tempered chromatic scale. We modify the original frequency scaling function defined in Equation 1 in two alternative ways that differ only in the choice of constants, as shown in Equations 2 and 3:

$$\begin{aligned} \tilde{f}_{\text{mel}} &= 1127 \cdot \ln(1 + f/700) \quad (1) \\ &= \ln(2) \cdot 1127 \cdot \log_2(1 + f/700) \\ \tilde{f}_{\text{notes}} &= 12 \cdot \log_2(f/440) \quad (2) \\ \tilde{f}_{\text{log2}} &= \log_2(1 + f). \quad (3) \end{aligned}$$

With $\tilde{f}_{\text{notes}}$, setting the number of filter bank channels to 40 and adjusting the lowest and the highest frequency centers to 196 Hz and 2093 Hz (which are only slightly different from the frequency range of the baseline front-end) results in a filter bank where the filter center frequencies are aligned with "note frequencies", i.e., $\{440 \cdot 2^{n/12}$ Hz $\mid -13 \leq n \leq 26\}$, thus covering roughly three octaves.

2.2. Acoustic Model Improvements

In addition to improving the quality of the features used by our system, we have also explored a number of modeling improvements. In all of our experiments, we use 512 music phonemes, a setting that has been empirically observed to attain an acceptable tradeoff between the granularity of sound clustered into a given phoneme and computational efficiency.

On average, each music phoneme has a duration of 6 frames; the duration follows approximately the Poisson distribution (with $\lambda = 6$). Each song from our collection is described by a sequence of approx. 22,000 phonemes on average, and each song's transcription uses almost every phoneme from the repository. The training starts with a single Gaussian acoustic model and proceeds to train a mixture model by splitting clusters of observations along the direction of maximal variance. Thus the phonemes represent short homogeneous sequences of feature vectors, a concept analogous to the meaning of "phoneme" in an ASR context. Figure 1 shows an example of an alignment plotted over the spectrogram of the corresponding audio segment.

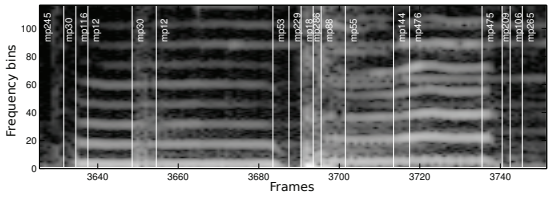

Fig. 1. Example of an alignment

Training the acoustic models solely on reference recordings results in an inherent mismatch between the models and test recordings produced with mobile phones in noisy settings. As described in Section 4, we have recorded our entire training data set with multiple mobile phone microphones with the goal of reducing this mismatch. However, pooling multiple recordings of the same song naïvely in the training stage allows for a clean and distorted recording of the same song to receive different transcriptions. In order to ensure that the transcriptions are the same amongst the different versions of a given song, training is initially performed from scratch on the clean corpus. Next, a second training pass performs a forced alignment of mobile recordings for each song to the same song's transcriptions obtained on clean audio. This type of *multi-conditional training* is beneficial not only because the acoustic models are prevented from overfitting the clean data, but also because this discourages the creation of disparate phoneme clusters for different recording conditions.

While using distorted training data in addition to clean recordings during training substantially improves mobile music recognition quality, this type of training can be vulnerable to overfitting the specific distortions observed in the training data. In order to improve the robustness of the acoustic models to previously unseen distortions, we add a small level of white Gaussian noise to the training data.

2.3. Duration modeling

In the baseline system, the recognition FST is compiled from transcriptions by introducing a single hidden Markov model (HMM) state per music phoneme, disregarding its actual length. This is a suitable approach for speech recognition, where variations in speaking rate are captured by HMM topology. In contrast, in music recognition, we do not expect changes of playback speed from one music phoneme to the next, and hence it is reasonable to model the duration of each phoneme explicitly. This can be accomplished by replicating a given HMM state once for each frame spanned by a given music phoneme at each point in the indexed song. We also disallow self-loops in the HMM, thus requiring the decoder to transition to a new HMM state on each acoustic frame. This has the effect of enforcing the constraint that for a given song, the number of frames seen for each particular music phone is exactly the same as that in the reference recording.

While this increases the size of the recognition FST approximately by a factor of 6 (in terms of number of arcs/states), it allows for much greater constraint during the decoding process, and thus, improved accuracy.

3. STRUCTURAL ANALYSIS

One advantage of model-based music recognition is that it provides a labeling of music audio in terms of elementary sound units. The approach presented here provides a symbolic representation that naturally enables a structural analysis of song audio. Such a representation has many applications, including automatic segmentation of a recording into high-level sections for tasks such as chorus detection, as well as automated identification of common melodic elements across songs to detect plagiarism or song covers. In addition, the representation of audio as a low-dimensional music phoneme sequence allows the use of statistical methods such as language modeling, bag-of-words analysis, approximate repeated substring matching, and many others. A comprehensive study of these techniques is beyond the scope of this paper; nevertheless, to motivate future work we present an initial analysis of song audio using music phoneme acoustic models.

Structural analysis of song audio has been previously studied in several works, where the predominant technique has been, as suggested by Foote [10], to compute a similarity measure between all pairs of feature vectors corresponding to the song audio. Several researchers have developed techniques comparing feature vectors [11, 12, 13]. Let $x_i, i \in [1, n]$ be the feature vectors computed for a song. Then the self-similarity matrix $\mathbf{A}$ is defined as $\mathbf{A}_{i,j} = \text{sim}(x_i, x_j)$ for some similarity score $\text{sim}(\cdot, \cdot)$. With such a representation of self-similarity, off-diagonal rectangles (p, q, r, s) where $\mathbf{A}_{i,j}, i \in [p, q], j \in [r, s]$ are large represent repeated sounds within the song. Analogously, block diagonal submatrices off the main diagonal correspond to a repeated sequence of sounds, such as chorus.

One drawback of such a metric stems from the fact that even if for some values of $\{i, j\}$, x_i and x_j represent a repeated segment of music, natural variations in polyphonic music performance are expected to make the similarity inside the rectangle $\mathbf{A}_{i,j}, i \in [p, q], j \in [r, s]$ quite noisy, rather than of a consistently high value. Our model-based representation of music audio allows the self-similarity to be computed on the level of the phonemes aligned to the feature vectors. Let the audio segments s_1, s_2 be the feature vector ranges $x_1 \in s_1$ and $x_2 \in s_2$ labeled with music phonemes p_1 and p_2, respectively. Then their similarity can be captured by a symmetrized approximation of the Kullback-Leibler divergence (KL) as $\text{sim}(s_1, s_2) = \text{KL}(G_1||G_2)$, where G_1 and G_2 are the GMMs corresponding to p_1, and p_2, respectively. Various approximations of KL divergence of GMMs are reviewed in [14]. This representation is advantageous firstly because the unification of same-phoneme segments in the audio greatly reduces the noise in the similarity signal, and also because it is no longer necessary to compute pairwise similarity between all frames in the song audio – rather, the similarity between all pairs of GMMs can be precomputed. Figure 2 shows a self-similarity matrix and the spectrogram of the corresponding audio segment of 3 seconds.

Name	Phone (distance)	# songs	# snippets	Dur. [h]
D1	Motorola Droid (close)	458	5799	24.1
D2	Motorola Droid (far)	457	5765	24.0
N1	Nexus One (far)	457	5765	24.0
N2	Nexus One (close)	93	1176	4.9

Table 1. Data sets

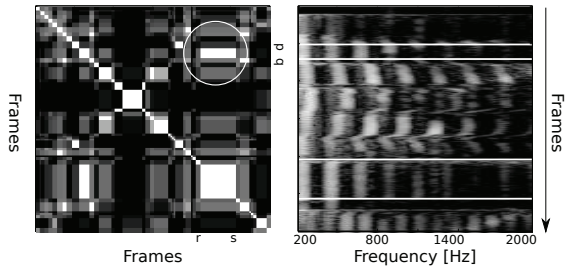

Fig. 2. Self-similarity matrix **A** (left) with aligned spectrogram **S** (right). The encircled block implies that the frames in $[p, q]$ have a high similarity to those in $[r, s]$, also seen in **S**.

4. EXPERIMENTAL RESULTS

Our training set consists of 5000 clean CD-quality song recordings (set CLEAN) of which we use 500 songs (set INDEX) to build our recognition WFST. For our modeling experiments, we augment the CLEAN training set with roughly 4500 songs (from CLEAN - INDEX) that are captured in full on two Motorola Droid phones (set RECORD). The test set was created by capturing the remaining songs (i.e. those in INDEX; call this set RECORDINDEX) on four phones in different locations relative to the signal source and splitting each song into 15 second non-overlapping snippets. After removing the first and last snippets from each song, which contain mostly silence, we obtain the four sets presented in Table 1. The total duration of the test snippets is 77 hours. These sets enable evaluating our recognition system's robustness to various non-linear distortions that are not sufficiently modeled by artificially adding noise to clean recordings. Note that our system is currently outputting a single-best hypothesis only, and hence we report 1-best accuracies rather than precision and recall. However, it would be possible also to modify our decoding setup to output an *n*-best list or a lattice of results.

Experiment	D1	D2	N1	N2	Avg.
20 filters, 125-3800 Hz (baseline)	25.7	83.1	78.9	63.4	62.6
24 filters, 200-2000 Hz, win. 0.5 s	73.9	80.2	89.7	63.6	80.1
scaling with $\tilde{f}_{\log 2}$	81.9	81.3	89.5	67.8	83.2
scaling with $f_{\log 2}$, no PLP	82.3	81.1	89.1	68.3	83.2
scaling with f_{Notes}, 40 filters, 196-2093 Hz, 15 cepstral coeff.	82.3	81.4	89.3	70.2	83.4
(as above) + white noise in training	84.4	84.2	92.2	71.3	85.9

Table 2. Accuracy for front-end adaptation experiments

Experiment	N1	N2	Avg.
1st pass (baseline)	92.2	71.3	88.7
2nd pass: unconstrained	92.9	75.4	89.9
2nd pass: clean transcriptions	93.2	76.4	90.4

Table 3. Accuracy for multi-conditional training

Front-end. We test the modifications we propose in Section 2.1 to our baseline speech recognition front-end (similar to that reported in [1]) by training an acoustic model on CLEAN, building the FST from the songs in INDEX, and testing the modifications on the four sets of snippets. As shown in Table 2 adapting the front-end to the task of music recognition consistently improved identification accuracy across the different test sets. Note that even though slightly higher accuracy can be obtained by increasing the GMM training iterations, we chose to stop at three iterations in all our experiments to avoid overfitting.

Acoustic Modeling. For the modeling experiments, we start with the best front-end from Table 2 and an acoustic model trained on CLEAN. We initiate a second training pass that force aligns the songs in RECORD and RECORDINDEX to the clean transcriptions as described in Section 2.2. Since the recorded sets are captured on Droid phones, we exclude sets D1 and D2 from the evaluation. It is noteworthy, however, that as expected the system achieves 100% accuracy on these sets. The results of our multi-conditional training are shown in Table 3, which also includes the accuracy of a system that does not constrain the transcriptions to be identical to those obtained in the first pass. Although the reported gain in accuracy is not big, the constrained system consistently outperformed the unconstrained one across all experiments.

Duration Modeling. We applied duration modeling to systems with various front-end configurations following the motivation in Section 2.3. The results in Table 4 show that duration modeling helps compensate for less robust front-ends.

5. CONCLUSION

We have described approaches to improve the robustness of music recognition and presented an evaluation on real cell phone recordings. We modified a speech recognition front-end to fit the needs of music recognition, and we introduced a technique for producing song transcriptions that accounts for the fact that the same song can be recorded under variable conditions. Our method of music phoneme duration modeling improved the recognition results consistently in all experiments. Finally, we adapted the technique of self-similarity for recognition of repeated patterns to our ASR-based system and obtained nicely interpretable results. We anticipate further quantitative analysis of this method in future work. We also plan to explore the possibility of using a true duration model - that is, one that represents the duration of a music phone with a probability distribution, to improve robustness to global playback rate changes.

Front-end	no DM	DM	rel. Δ [%]
baseline, window 0.5 s	75.3	80.5	6.9
scaling with $\tilde{f}_{\log 2}$	81.9	83.9	2.4
scaling with $\tilde{f}_{\text{Notes}}$, no PLP	84.2	84.5	0.4

Table 4. Accuracy for duration modeling (DM) on set D1

6. ACKNOWLEDGMENTS

We would like to thank Matt Sharifi, Annie Chen, David Talkin, Dick Lyon, Tom Walters, Pedro Moreno, Hank Liao and Michiel Bacchiani for many useful discussions.

7. REFERENCES

[1] M. Mohri, P. Moreno, and E. Weinstein, "Efficient and robust music identification with weighted finite-state transducers," in *IEEE Trans. on Audio, Speech and Language Processing*, 2010, vol. 18(1).

[2] E. Weinstein and P. Moreno, "Music identification with weighted finite-state transducers," in *Proc. of ICASSP*, Honolulu, HI, USA, April 2007.

[3] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. of the 25th Int. Conf. on Very Large Data Bases*, San Francisco, CA, USA, September 1999, VLDB '99, pp. 518–529, Morgan Kaufmann Publishers Inc.

[4] S. Baluja and M. Covell, "Audio fingerprinting: Combining computer vision and data-stream processing," in *Proc. of ICASSP*, Honolulu, HI, USA, April 2007.

[5] A. Wang, "An industrial-strength audio search algorithm," in *Proc. of ISMIR*, Baltimore, MD, USA, October 2003.

[6] M. Mohri, P. Moreno, and E. Weinstein, "Robust music identification, detection, and analysis," in *Proc. of ISMIR*, Vienna, Austria, September 2007.

[7] B. Logan, "Mel frequency cepstral coefficients for music modeling," in *Proc. of ISMIR*, Plymouth, MA, USA, October 2000.

[8] J. Schalkwyk et. al., "Google search by voice: A case study," in *Advances in Speech Recognition: Mobile Environments, Call Centers and Clinics*, chapter 4, pp. 61–90. Springer, 2010.

[9] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *Journal of the Acoustical Society of America*, vol. 87(4), pp. 1738–1752, 1990.

[10] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *Proc. IEEE Int. Conf. on Multimedia and Expo*, New York, NY, USA, August 2000, pp. 452–455.

[11] W. Chai, *Automated Analysis of Musical Structure*, Ph.D. thesis, MIT, 2005.

[12] M. A. Bartsch and G. H. Wakefield, "To catch a chorus: Using chroma-based representations for audio thumbnailing," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, USA, October 2001.

[13] R. B. Dannenberg and N. Hu, "Discovering musical structure in audio recordings," in *Music and Artificial Intelligence: Second International Conference; LNCS vol. 2445*, Berlin, 2002, pp. 43–57, Springer.

[14] J. R. Hershey and P. A. Olsen, "Approximating the Kullback Leibler divergence between Gaussian mixture models," in *Proc. of ICASSP*, Honolulu, HI, USA, April 2007.