# MUSIC IDENTIFICATION WITH WEIGHTED FINITE-STATE TRANSDUCERS

*Eugene Weinstein* [1,2] *and Pedro Moreno* [1]

[1] Google Inc.
76 Ninth Avenue, New York, NY 10011.
[2] Courant Institute of Mathematical Sciences
251 Mercer Street, New York, NY 10012.

## ABSTRACT

Music identification is the process of matching an audio stream to a particular song. Previous work has relied on hashing, where an exact or almost-exact match between local features of the test and reference recordings is required. In this work we present a new approach to music identification based on finite-state transducers and Gaussian mixture models. We apply an unsupervised training process to learn an inventory of music phone units similar to phonemes in speech. We also learn a unique sequence of music units characterizing each song. We further propose a novel application of transducers for recognition of music phone sequences. Preliminary experiments demonstrate an identification accuracy of 99.5% on a database of over 15,000 songs running faster than real time.

***Index Terms***— Music identification, acoustic modeling, finite-state transducers.

## 1. INTRODUCTION

Automatic identification of music has recently been of substantial interest [4, 1, 3]. This technology can be applied to a number of usage scenarios. End users can identify the song title, album and recording artist(s) of a song with just a short audio snippet. Content distribution networks can identify copyrighted audio within their systems. Finally, recording labels can monitor radio broadcasts to ensure correct accounting. Music identification is challenging because the recording might be noisy, of poor quality, and/or marred with channel effects. In addition, the test recording might consist of only a few seconds of audio which needs to be aligned to the reference recording to make a match.

Previous work in music identification can be classified into hashing and search approaches. Hashing approaches (e.g., [4]) compute local fingerprints for an audio snippet, retrieve songs with matching fingerprints from a hash table, and pick amongst these candidates using some metric of match accuracy. Some hashing approaches [3] learn the features discriminatively. In a search-based approach [1], cepstral features over the audio stream are decoded directly into a sequence of audio events, as in speech recognition. Both the decoding and the mapping of sound sequences to songs is driven by a hidden Markov model (HMM) representation. However, this system looks only for atomic sound sequences of a particular length, presumably to control search complexity.

In this work we present a search based approach using finite-state transducers (FSTs). We learn an inventory of music sound units via clustering, and train Gaussian mixture models for each unit. The use of Gaussian mixtures to model the acoustics of music phones allows us to leverage techniques commonly used in speech recognition. Our acoustic modeling approach can tolerate small variations in acoustic conditions more naturally than hashing, which is based on exact or almost exact matching of fingerprints. In our approach songs are represented by unique sequences of music phones. We construct a compact recognition transducer where each path corresponds to a unique song. This representation offers many unique advantages. For instance generic automata operations such as minimization allow us to remove redundancies in the song graph. As a result efficient search is possible. Effectively we propose to transform the problem of music identification into a string matching operation where FSTs are a natural formalism for constraining the search.

## 2. ACOUSTIC MODELING

Our acoustic modeling approach consists of jointly learning an inventory of *music phones* and the sequence of units best representing each song. We represent each song as a sequence of mel-frequency cepstral coefficient (MFCCs) vectors. Cepstra have been used extensively in speech processing, and have also recently been shown to be effective in the analysis of music [1, 9]. We use the first twelve coefficients, the energy, and their first and second derivatives to produce a 39-dimensional feature vector.

Each song is initially broken into pseudo-stationary segments. We use 100ms windows over the feature stream. Single diagonal covariance Gaussian models are fitted to each window and the Kullback-Liebler (KL) divergence is measured between adjacent windows. We hypothesize segment boundaries where the KL divergence is above an experimentally determined threshold.

We next apply clustering to the segment collection. For each segment we train a single initial diagonal covariance Gaussian model. As in [5] we use maximum likelihood as an objective distance function rather than the more common Euclidean distance. See [5] for further details.

Since song data labeled with music phone transcriptions is not available we cannot use the standard EM training al-

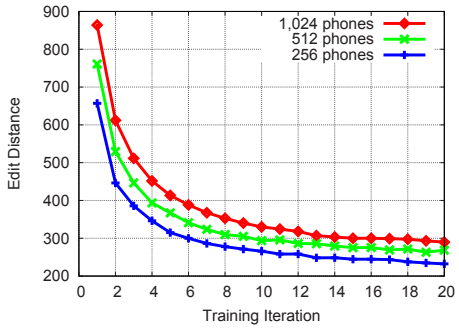



**Fig. 1**. Average edit distance per song vs. training iteration

gorithm. Instead we use an approach similar to that of [1]. Music phones are represented with Gaussian mixture acoustic models (GMMs). We alternate between finding the best music phone transcription per song given the current model and refining the GMMs given the current transcriptions.

The reference transcription changes for each iteration of GMM learning. Therefore it is not possible to use training data likelihood as a measure of convergence. We instead use the edit distance, a string similarity metric defined for our purposes as the minimal number of insertions, substitutions, and deletions of music phones required to transform one transcription into another. For a song set $S$ let $t_i(s)$ be the transcription given to song $s$ at iteration $i$ and $\mathrm{ED}(a, b)$ the edit distance of sequences $a$ and $b$. At each iteration $i$, we compute the total edit distance $C_i = \sum_{s \in S} \mathrm{ED}(t_i(s), t_{i-1}(s))$ as our convergence measure. We have found $C_i$ to converge after around twenty iterations. Figure 1 illustrates how this quantity changes during training for three phone inventory sizes.

## 3. RECOGNITION TRANSDUCER

Our music identification system is based on weighted finite-state transducers and Viterbi decoding as is common in speech recognition [8]. The decoding is based on the acoustic model described in the previous section and a compact transducer that maps music phone sequences to corresponding song identifiers. This section describes the algorithms for constructing such a transducer.

Given a finite set of songs $S$, the music identification task consists of determining the songs in $S$ that contain a query song snippet $x$. Since a song snippet may be an arbitrary sample extracted from a song, the recognition transducer must map any sequence of music phones appearing in a song to the corresponding song identifiers.

### 3.1. Factor automata

More formally, let $\Delta$ denote the set of music phones. The song set $S = \{x_1, \ldots, x_m\}$ is a set of sequences in $\Delta^*$. A *factor*, or *substring*, of a sequence $x \in \Delta^*$ is a sequence of consecutive phones appearing in $x$. Thus, $y$ is a factor of $x$ iff there exists $u, v \in \Delta^*$ such that $x = uyv$. The set of factors of $x$ is denoted by $\mathrm{Fact}(x)$ and more generally the set of factors of all songs in $S$ is denoted by $\mathrm{Fact}(S)$. A correct transcription of an in-set song snippet is thus an element of $\mathrm{Fact}(S)$.

In what follows we will assume that only one song identifier $s_x$ is associated to each factor $x$. This in fact turns out to be empirically the case even for relatively short song snippets. The recognition transducer $T$ must thus represent the following mapping:

$$\begin{array}{rccl} [\![T]\!] : \mathrm{Fact}(S) & \to & \mathbb{N} \\ x & \mapsto & [\![T]\!](x) = y_x. \end{array} \quad (1)$$

The size of $\mathrm{Fact}(S)$ may be quite large, and thus it is not immediately clear that it is possible to construct a compact transducer with this property. In our experiments, $|S|$ is about 15,000 and the average length of a song is more than 1,700. Thus, in the worst case, $|\mathrm{Fact}\, S|$ can be as large as $15{,}000 \times 1{,}700^2 \approx 43 \times 10^9$. The size of a naive trie-based representation would thus be prohibitive.

However, several automata-theoretic results suggest that a compact representation may be possible [2]. In fact, the size of the minimal deterministic automaton representing a single sequence $x$ is linear in $|x|$. More precisely [2]:

$$|Q| \leq 2|x| - 2 \qquad |E| \leq 3|x| - 4. \quad (2)$$

where $Q$ and $E$ are the set of states and transitions of the factor automaton of $x$. For the case of the minimal deterministic factor automaton representing $S$, we have found that the number of transitions is about twice that of the original automaton. In particular, the minimal automaton representing all 15,000 songs has about 27.5M transitions and the corresponding minimal factor automaton about 59.5M transitions.

### 3.2. Factor Transducer Construction

In the following, we describe the construction of the factor automaton of $S$ from a finite automaton accepting the sequences of $S$, and more generally, the construction of a compact recognition transducer $T$.

We construct a deterministic and minimal automaton representing the sequences in $S$ and more generally a deterministic (subsequential) and minimal finite-state transducer mapping each song to its identifier using transducer determinization and minimization algorithms [6, 7]. Figure 2 shows the transducer mapping each song to its identifier before application of determinization and minimization, when $S$ is reduced to three short songs.

Let $A_0$ be the acceptor obtained by omitting the output labels of $T_0$. The factor automaton $A$ of $S$ is constructed from $A_0$ by creating an $\epsilon$-transition from the initial state of $A_0$ to all other states of $A_0$ and making all states of $A_0$ final. This automaton clearly accepts any factor of $A_0$. Applying determinization and minimization, we create the minimal deterministic automaton $A_1$ representing $\mathrm{Fact}(S)$. Figure 3(a) shows the result of that construction when applied to the acceptor $A_0$ obtained by omitting the output labels of the transducer of Figure 2.

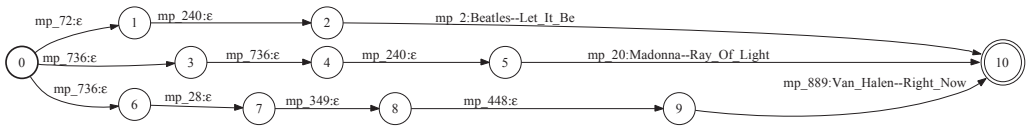
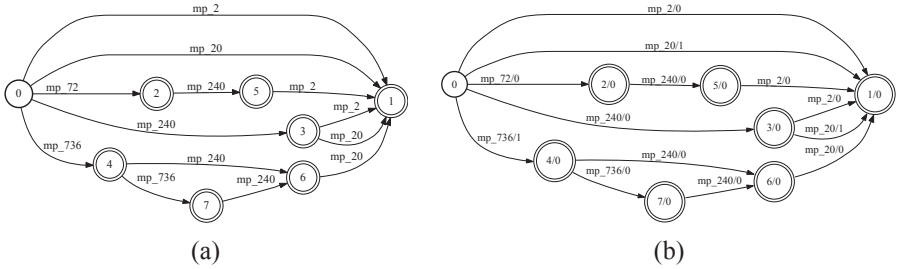



**Fig. 2**. Finite-state transducer $T_0$ mapping each song to its identifier.




(a) (b)

**Fig. 3**. (a) Deterministic and minimal unweighted factor acceptor $A_1$ for two songs. (b) Deterministic and minimal weighted factor acceptor $A_2$ for two songs.

This leads to a compact automaton representing $\mathrm{Fact}(S)$, but it does not allow us to output the song identifier associated with each factor. To accomplish this, one could augment all accepting paths of the automaton $A_1$, i.e. all factors, with output labels corresponding to the song identifiers. However, since output labels cannot be shared among factors along the same path, this would require splitting states to separate paths and would yield a transducer exponentially larger than $A_1$.

We instead create a compact weighted acceptor over the *tropical semiring* accepting $\mathrm{Fact}(S)$ that associates the total weight $s_x$ to each $x \in \mathrm{Fact}(S)$. A crucial advantage of this representation is the use of weighted determinization and minimization [6] during which the song identifier is treated as a weight that can be distributed along a path. The property that the sum of the weights along the path labeled with $x$ is $s_x$ is preserved by these operations.

$A_2$ is constructed by adding weights to $A_1$. The weight assigned to an $\epsilon$-transition reaching state $q$ is the integer song identifier of the unique accepting path in $A_0$ going through $q$. All other weights are zero. This guarantees the weight of each factor $x$ to be exactly $s_x$. The final automaton $A_2$ is produced by applying weighted determinization and minimization over the tropical semiring. The weighted acceptor $A_2$, obtained after determinization and minimization, is transformed into a transducer $T$ by simply treating each output weight integer as a regular output symbol.

Figure 3(b) shows the weighted automaton $A_2$ corresponding to the unweighted automaton $A_1$ of Figure 3(a). Note that the number of states and transitions has not increased from $A_1$ to $A_2$. Remarkably, even in the case of 15,000 songs, the total number of transitions of the weighted acceptor $A_2$ was about 59.6M, only about $0.17\%$ more than that of $A_1$. This confirms empirically the benefits of our representation. Thus, the construction algorithm just described leads to a recognition transducer $T$ whose size is less than 2.2 times that of the minimal deterministic transducer representing all songs.

### 3.3. Weighted Factor Transducers

The finite-state transducer $T$ just described is unweighted, that is all song snippets are assigned the same log-likelihood. However the distribution of the weights along a given path can be changed (without affecting the total weight) to substantially improve recognition speed when using a Viterbi decoder with beam pruning. This can be done by applying the weight pushing algorithm in the *log semiring* [7] to the transducer $T$. As a result each prefix $x$ of a path in $T$ is weighted according to the frequency of snippets starting with $x$. As a result, more frequent factors are preferred during decoding. Our experiments show that this can result in a substantially higher accuracy for a given speed.

## 4. EXPERIMENTAL RESULTS

Our data set consisted of $15{,}455$ songs in 128kbps MP3 format. The average song duration was 3.9 minutes, for a total of over $1,000$ hours of training audio. For the main set of experiments, we trained and tested on "clean" data, i.e., audio without any added noise or distortions. However, a preliminary experiment on $1,000$ song snippets mixed with additive white noise at a SNR of 25dB resulted in an identification accuracy of $95.9\%$, suggesting that our acoustic models are robust to noise.

Figure 4 gives accuracy rates for three experiments. All figures are for in-set song classification only – the rejection performance of the classifiers is tested separately. We selected $15{,}455$ randomly collected segments (one per song) from our song collection. FS is the full song identification task where we use a full song transducer $T_0$ and full song snippets. This experiment tests the performance of the system under ideal

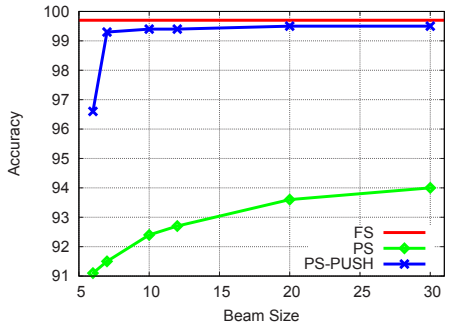



**Fig. 4**. In-set accuracy for three different experiments

circumstances. PS is the partial song task where identification is performed on ten seconds of song audio, with the unweighted factor transducer $T$. PS-PUSH is the same task after using weight pushing to add weights to $T$.

We have also tested the ability of our system to reject out-of-set songs. We constructed a mixture Gaussian background acoustic model by clustering mixture components across all music phones. We then constructed a three-dimensional feature vector $[L_r, L_b, (L_r - L_b)]$ for each song snippet, where $L_r$ and $L_b$ are the log-likelihoods of the best path and background acoustic models, respectively. We applied a support vector machine (SVM) to $15{,}455$ positive and $13{,}811$ negative (out-of-set) song snippets. With ten-fold cross-validation and a radial basis function kernel, we achieve an accuracy of $96.4\%$.

Music identification using the whole song audio works almost flawlessly ($99.7\%$ in-set identification accuracy). Ten second snippets are recognized with the factor transducer with degraded accuracy (low $90$'s). However, weight pushing improves the accuracy of the system back up to $99.5\%$. All experiments run faster than real time: for instance with a search beam width of $12$, the runtime is $0.48$ of real time.

## 5. CONCLUSION

In this paper we describe a novel application of weighted finite-state transducers to music identification. In contrast to previous approaches our system does not rely on obtaining a near-exact match between test and reference feature values. In addition our system matches music phone sequences of arbitrary length. We have performed an initial set of identification experiments which have demonstrated the accuracy and potential of our system.

We believe that the combination of GMMs for acoustic modeling and FSTs for song structure representation and manipulation offers some intriguing advantages over more traditional hashing schemes. GMMs naturally tolerate small changes in the music signal. Simple signal processing techniques such as cepstral mean and variance normalization can be also leveraged to handle small noise and channel distortion effects. Our approach also learns a symbolic representation for each song, i.e., how to transcribe each song as a sequence of music units. While we have not explored this aspect deeply, preliminary experiments suggest that this representation may yield interesting insights into structural features of music, such as repeated chorus elements and tune similarity, or plagiarism, across songs.

In future work, we will characterize the performance of the system in the presence of various types of noise and distortions. We also plan to explore the use of discriminative learning techniques to further improve the performance of the system under noise and signal distortion conditions.

### Acknowledgements

The authors would like to sincerely thank Mehryar Mohri for his substantial contribution to this project. In addition, we would like to acknowledge all the members of the Google speech team, especially Michiel Bacchiani, Johan Schalkwyk, and Michael Riley, for their help and advice. This work was partially funded by NYSTAR and by the Dept. of the Army Award Number W23RYX-3275-N605. The U.S. Army Medical Research Acquisition Activity is the awarding and administering acquisition office. The content of this material does not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred.

## 6. REFERENCES

[1] E. Batlle, J. Masip, and E. Guaus. Automatic song identification in noisy broadcast audio. In *IASTED International Conference on Signal and Image Processing*, Kauai, Hawaii, 2002.

[2] M. Crochemore. Transducers and repetitions. *Theoretical Computer Science*, 45(1):63–86, 1986.

[3] M. Fink, M. Covell, and S. Baluja. Social and interactive television application based on real time ambient audio indentification. *EuroITV 2006*, May 2006.

[4] J. Haitsma, T. Kalker, and J. Oostveen. Robust audio hashing for content identification. In *Content-Based Multimedia Indexing (CBMI)*, Brescia, Italy, September 2001.

[5] M.Bacchiani and M. Ostendorf. Joint lexicon, acoustic unit inventory and model design. *Speech Communication*, 29:99–114, November 1999.

[6] M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.

[7] M. Mohri. Statistical Natural Language Processing. In M. Lothaire, editor, *Applied Combinatorics on Words*. Cambridge University Press, 2005.

[8] M. Mohri, F. C. N. Pereira, and M. Riley. Weighted Finite-State Transducers in Speech Recognition. *Computer Speech and Language*, 16(1):69–88, 2002.

[9] D. Pye. Content-based methods for the management of digital music. In *ICASSP*, pages 2437–2440, Istanbul, Turkey, June 2000.