On Musical Performances Identification, Entropy and String Matching

Antonio Camarena-Ibarrola and Edgar Chávez

Universidad Michoacana de Sán Nicolás de Hidalgo
Edif "B" Ciudad Universitaria CP 58000
Morelia, Mich., México
{camarena,elchavez}@umich.mx

Abstract. In this paper we address the problem of matching musical renditions of the same piece of music also known as *performances*. We use an entropy based Audio-Fingerprint delivering a framed, small footprint AFP which reduces the problem to a string matching problem. The Entropy AFP has very low resolution (750 ms per symbol), making it suitable for flexible string matching.
We show experimental results using dynamic time warping (DTW), Levenshtein or *edit* distance and the Longest Common Subsequence (LCS) distance. We are able to correctly (100%) identify different renditions of masterpieces as well as pop music in less than a second per comparison. The three approaches are 100% effective, but LCS and Levenshtein can be computed online, making them suitable for monitoring applications (unlike DTW), and since they are distances a metric index could be use to speed up the recognition process.

1 Introduction

The alignment of musical performances has been a subject of interest in several *Music Information Retrieval* disciplines such as *Polyphonic Audio Matching* [1], *querying by melody* [2] and *score-performance matching* [3], or its on line version called *score-performance following* [4]. The last discipline has the goal of qualifying where a performance is in respect to a score, thus enabling automatic accompaniment and automatic adding of special effects based on the position of the performance in time, according to meta-data included in the score.

In this paper we propose a method for comparing performances allowing online detection of occurrences in an audio channel, for this purpose, we make use of efficient and versatile aligning techniques developed for matching DNA sequences [5] and for finding strings occurrences in texts allowing errors [6]. Tests using the classical DTW technique were also included as a reference. Hidden Markov Models (HMM) were not considered in the experiments due to the fact that they need training to compute their optimal parameters and topology which has to be done at designing time, if the collection of songs changed, the topology would not be optimal any more, redesigning the HMM every time a song is added to the collection is impractical especially if the HMM has already been implemented in

hardware (i.e Field Programmable Gate Arrays), therefore, using HMM as an aligning technique was discarded.

For the feature extraction level, a string AFP based on an Information Content Analysis was preferred since it has probed to be very robust to signal degradations although never tried in matching musical performances [7], also because it is a string AFP of short length being the outcome of a low resolution analysis.

AFPs are compact content based representations of audio files which must be robust to common signal degradations like noise contamination, equalization, lossy compression and re-recording (Loudspeakers to microphone transmission). Existing AFPs are basically of four kinds: (I) *Sequences of Feature Vectors* also known as *trajectories* or *traces* since they are extracted at equally spaced periods of time, an example of this is the spectral flatness based AFP used by MPEG-7 [8]. (II) *Single vectors* are the smallest AFPs, they usually include the means and variances of features extracted from the whole song, (i.e Beats per Minute) this AFPs do not require any aligning technique but are not very robust to signal degradations. (III) *Strings* resulting from codification of feature vectors. Haitsma-Kalker "Hash string" [9] or the entropy-based AFP used in this work [7] are good examples of this kind of AFPs. (IV) HMMs are also used as AFPs, normally a HMM is built for each one of the songs from the collection [10].

For the purpose of matching performances, *trajectories* AFPs are suitable for DTW, however, to compare them using flexible string matching techniques, they would have to be subject to some vector quantization technique in order to turn them into strings, this implies some precision loss. *String* AFPs are suitable for matching performances using flexible string matching distances, Haitsma-Kalker's AFPs are extremely long strings since they were designed to identify songs with only 3 seconds of audio and therefore result from a very high resolution analysis, however they are impractical in matching performances for the high computational cost needed for aligning them. For example, a 5 minute song would be represented by a string whose length would be of 25 862 characters each one from an alphabet of 2^{32}. Finally, since the spectral entropy based AFP is obtained from a low resolution analysis a 5 minute song will produce a string of only 400 characters from an alphabet of 2^{24}, quite suitable for our problem.

1.1 The spectral entropy based audio fingerprint

Claude Shannon stated that the level of *information* in a signal could be measured with Boltzman's formula (1) for computing entropy in gases which as we know is a measure of chaos or disorder [11].

$$H(x) = E[I(p)] = \sum_{i=1}^{n} p_i I(p) = -\sum_{i=1}^{n} p_i ln(p_i) \tag{1}$$

Entropy has been used in speech signals as a segmentation criterium in noisy environments [12] and in deciding the desirable frame rate in the analysis of Speech signals [13], but it had never been used as the main feature used for matching audio files. Recently, a first Entropy-based AFP was proposed in [14]

which estimates the *information content* in audio signals every second directly in time domain using histograms, later the same authors proposed an espectral entropy based AFP [7] which is more robust to noise, equalization and loudspeaker to microphone transmission than the spectral flatness based AFP adopted by MPEG-7 [15]. The espectral entropy based AFP results from the codification of the sign of the derivative in time of the entropy for critical bands 1 to 24 according to Bark scale. The signal is first segmented in frames of 1.5 seconds with 50 percent overlapping so that one vector of 24 ceros and ones is obtained every 750 milliseconds, to every frame, a Hanning window is applied, then taken to the frequency domain via the fast fourier transform (FFT). For every critical band, the result is considered to be a two dimensional (i.e. real and imaginary part), random variable with gaussian distribution and mean zero, according to [16]. The spectral entropy for band p is determined using equation (2)

$$H = ln(2\pi e) + \frac{1}{2} ln(\sigma_{xx}\sigma_{yy} - \sigma_{xy}^2) \tag{2}$$

where σ_{xx} and σ_{yy} also known as σ_x^2 and σ_y^2 are the variances of the real and the imaginary part respectively and $\sigma_{xy} = \sigma_{yx}$ is the covariance between the real and the imaginary part of the spectrum in its rectangular form and so $\sigma_{xy}\sigma_{yx} = \sigma_{xy}^2$

Just as a spectrogram indicates the amount of energy a signal has both on time and frequency, a *entropygram* show the information level for every critical band and frame position in time. Figure 1 shows the entropygram of two performances of Mozart's *Serenade Number 13 Allegro* and figure 2 shows the entropygrams of two performances of Tchaikovsky's *Nutcracker waltz of the Flower*.

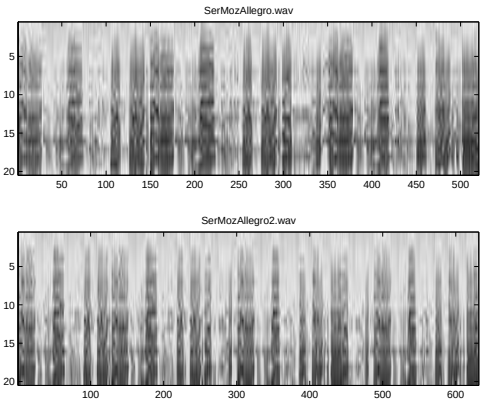

Fig. 1. Entropygrams of the two performances of Mozart's *Serenade Number 13 Allegro*

The sign of the Entropygram's time derivative is coded to built the string AFP as indicated in equation (3) where the bit corresponding to band b and

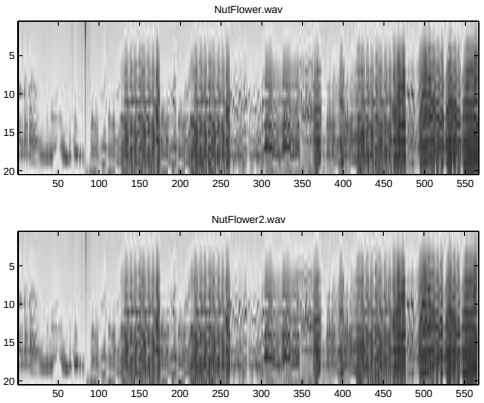

Fig. 2. Entropygrams of the two performances of Tchaikovsky's *Nutcracker waltz of the flower*

frame n (i.e. $b(n,b)$) is determined with the sign of the difference of the entropygram's entries $H(n,b)$ and $H(n-1,b)$. This is the same as coding the information of wether the entropy for each band is increasing or not.

$$\begin{aligned} b(n,b) = 1 \quad & \forall \quad H(n,b) - H(n-1,b) > 0 \\ = 0 \quad & \forall \quad H(n,b) - H(n-1,b) \leq 0 \end{aligned}$$

The spectral entropy based AFP can be seen as a string formed with symbols of 24 bits so that the alphabet size is 2^{24} and the number of symbols equals the duration of the musical performance in seconds multiplied by 4/3.

1.2 DTW

Aligning two performances $R(n),\ 0 \leq n \leq N$ and $T(m),\ 0 \leq m \leq M$ is equivalent to finding a warping function $m = w(n)$ that maps indices n and m so that a time registration between the time series is obtained. Function w is subject to the boundary conditions $w(0) = 0$ and $w(N) = M$ and might be subject to local restrictions, an example of such restriction is that if the optimal warping function goes through point (n,m) it must go through either $(n-1,m-1)$, $(n,m-1)$ or $(n-1,m)$ as depicted in figure 3, a penalization of 2 is charged when choosing $(n-1,m-1)$ and of 1 if $(n,m-1)$ or $(n-1,m)$ are chosen, this way the three possible paths from $(n-1,m-1)$ to (n,m) (i.e. first to $(n,m-1)$ and then (n,m)) will all have the same cost of 2. Other local restrictions defined by Sakoe and Chiba [17] can be used.

Let $d_{n,m}$ be the distance between frame n of performance R and frame m of performance T, then the optimal warping function between R and T is defined by the minimum accumulated distance $D_{n,m}$ as in (3).

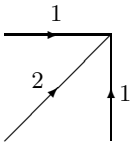

Fig. 3. Symmetric local restriction of first order

$$D_{n,m} = \sum_{p=1}^{n} d_{R(p),T(w(p))} \tag{3}$$

Once a local restriction is selected, $D_{N,M}$ can be computed using the recurrence defined in equations (4),(5) and (6), which correspond to local restriction shown in figure 3. Based on this recurrence $D_{N,M}$ can be efficiently obtained using dynamic programming.

$$D_{i,0} = \sum_{k=0}^{i} d_{i,0} \tag{4}$$

$$D_{0,j} = \sum_{k=0}^{j} d_{0,j} \tag{5}$$

$$D_{i,j} = min \begin{cases} D_{i-1,j-1} + 2d_{i,j} \\ D_{i-1,j} + d_{i,j} \\ D_{i,j-1} + d_{i,j} \end{cases} \tag{6}$$

1.3 String distances

The Levenshtein distance between two strings is defined as the number of operations needed to convert one of them into the other, the considered operations are *inserts*, *deletes* and *substitutions* and sometimes *transpositions*, a different cost to each operation may be considered depending on the specific problem. If only substitutions are allowed with the cost of 1, the distance is the same as the Hamming distance, if only insertions and deletions are allowed both with the cost of 1 the distance is known as the *Longest common subsequence* (LCS) distance, finally if only insertions are allowed at the cost of 1, the asymmetric *Episode distance* is obtained [6].

To compute the Levenshtein distance between the string t of length N and the string p of length M the equations (7),(8) and (9) are used assuming all edit operations (insert,delete and sustitutions) have the same cost of 1.

$$C_{i,0} = i \qquad \forall \quad 0 \leq i \leq N \tag{7}$$

$$C_{0,j} = j \qquad \forall \quad 0 \leq j \leq M \tag{8}$$

$$C_{i,j} = \begin{cases} C_{i-1,j-1} & t_i = p_j \\ min[C_{i-1,j-1}, C_{i,j-1}, C_{i-1,j}] + 1 & t_i \neq p_j \end{cases} \tag{9}$$

The classical approach for computing the Levenshtein distance relies in Dynamic Programming, for instance, the Levenshtein distance between the string "hello" and the string "yellow" is 2 as can be seen on location $(5, 6)$ of matrix (10) corresponding to two operations (i.e substitute "h" by a "y" and add a "w" at the end)

		y	e	l	l	o	w
	0	1	2	3	4	5	6
h	1	**1**	2	3	4	5	6
e	2	2	**1**	2	3	4	5
l	3	3	2	**1**	2	3	4
l	4	4	3	2	**1**	2	3
o	5	5	4	3	2	**1**	**2**

(10)

There is no need for keeping the whole dynamic programming table in memory. The Levenshtein distance can be computed maintaining only one column. To do so, initialize this only column with $C_i = i$ and then use equation (11) to update it while reading the text.

$$C_i^{'} = \begin{cases} C_{i-1} & t_i = p_j \\ min[C_{i-1}, C_{i-1}^{'}, C_i] + 1 & t_i \neq p_j \end{cases} \tag{11}$$

Where $C^{'}$ is the column being computed and C is the previous one

For the purpose of finding occurrences of a pattern on a text we must allow a match to occur at any time, this is achieved by setting $C_0^{'} = 0$ and monitoring if the last element on every column is less or equal to the predefined maximum distance. In matrix (12) the string *att* is found at positions 2, 3 and 5 with one error (i.e. substrings *at*, *atc*) and position 6 without errors inside the text *atcatt*.

		a	t	c	a	t	t
	0	0	0	0	0	0	0
a	1	0	1	1	0	1	1
t	2	1	0	2	1	0	1
t	3	2	**1**	**1**	2	**1**	**0**

(12)

2 Experiments

The original goal for this work was to test the spectral entropy based AFP described in section 1.1 in the problem of matching musical performances and

see how well this features could be aligned with a traditional techniques like DTW. However, using efficient and versatile aligning techniques commonly used in matching DNA sequences allows finding occurrences of music by any of its performances in an audio channel, which could not be done with DTW since it would require having both whole songs prior to the aligning. However DTW was included in the experiments as a reference to mesure the aligning capabilities of the Levenshtein and the LCS distances.

2.1 Using the Longest Common Subsequence Distance

To use the spectral entropy based AFP as a string, the symbols are considered to belong to an alphabet that is too large (2^{24}), it would be naive to consider two symbols as completely different just because they differ in one bit, remember that the symbols are made of bits that result from an information content analysis on unrepeatable audio segments, therefore, we considered two symbols as different only if the Hamming distance was grater than 7.

Once defined the rule to decide wether two symbols are different or not, the LCS distance will be used hopping that the same sequence of acoustic events will be present in both performances, only shorter subsequences in one of them with respect to the other, in this context a symbol represents an acoustic event.

The recurrence defined in (13),(14) and (15) is used with dynamic programming to compute the LCS distance.

$$C_{i,0} = i \qquad \forall \quad 0 \leq i \leq N \tag{13}$$

$$C_{0,j} = j \qquad \forall \quad 0 \leq j \leq M \tag{14}$$

$$C_{i,j} = \begin{cases} C_{i-1,j-1} & t_i = p_j \\ min[C_{i,j-1}, C_{i-1,j}] + 1 & t_i \neq p_j \end{cases} \tag{15}$$

2.2 Using the Levenshtein distance

Using a threshold to decide wether an acoustic event equals another may seem dangerous, so instead of throwing away the differences between the symbols, they may be used as the substitution cost in the Levenshtein distance while keeping the insertion and deletion cost to 1.

$$C_{i,j} = min \begin{cases} C_{i-1,j-1} + d(t_i, p_j) \\ C_{i,j-1} + 1 \\ C_{i-1,j} + 1 \end{cases} \tag{16}$$

Where $d(t_i, p_j) = Hamming(t_i, p_j)/24$ since t_i and p_j are made from 24 bits and we want $d(t_i, p_j)$ to be a value between 0 and 1.

2.3 Using DTW

Using local restriction depicted in figure 3, DTW was implemented with dynamic programming based on the recurrences defined with equations (4),(5) and (6) with $d_{i,j}$ being the Hamming distance between row i of one performance's AFP and row j of the other performance's AFP.

2.4 Normalizing the distances

The Levenshtein distance between performances of length N and M can not be greater than the length of the longest one of them, so in order to normalize the Levenshtein distance it was divided by $max(N, M)$, once normalized, it was posible to set a threshold to decided wether two performances match or not. The LCS distance can not be grater than $N + M$, so in order to normalize the LCS distance it was divided by $N + M$. The DTW distance was also divided by $N + M$.

2.5 The test set

Pairs of Master pieces from Mozart and Tchaikovsky played by different orchestras as well as pairs of beatles's songs played at two different events formed the test set of 40 audio files, the set of pairs is listed on table 1 where for each pair the duration of both performances is shown.

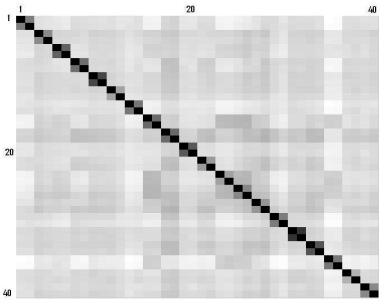

Fig. 4. Confusion Matrix result from using DTW

2.6 Results

The 40 audio files of the test set were put into comparison against each other, the 1 600 resulting distances were stored in a confusion matrix and represented as gray tones in figures 4, 5 and 6. A low distance is represented as a dark gray tone and a high distance as a light gray tone, the first raw have the distances between the first audio file and the rest of them, the second raw are the distances

Table 1. Pairs of Performances for the experiments. Performers: (i) *The Beatles*. (ii) *London Festival Orchestra, Cond.: Henry Adolph*. (iii) *London Festival Orchestra, Cond.: Alberto Lizzio*. (iv) *Camerata Academica, Cond.: Alfred Scholz*. (v) *Slovak Philharmonic Orchestra, Cond.: Libor Pesek*. (vi) *London Philharmonic Orchestra, Cond.: Alfred Scholz*.

Name	dur1	dur2
All my loving	2:09 (i)	2:08 (i)
All you need is love	3:49 (i)	3:46 (i)
Come together	4:18 (i)	4:16 (i)
Eleanor Rigby	2:04 (i)	2:08 (i)
Here comes the sun	3:07 (i)	3:04 (i)
Lucy in the sky with diamonds	3:27 (i)	3:27 (i)
Nowhere man	2:40 (i)	2:44 (i)
The Nutcracker Waltz of the flowers	7:09 (ii)	7:06 (iii)
The Nutcracker Dance of the Reeds	2:39 (ii)	2:41 (iii)
Octopus's garden	2:52 (i)	2:48 (i)
Mozart's Serenade 13 Menuetto	2:19 (iv)	2:10 (v)
Mozart's Serenade 13 Allegro	6:30 (iv)	7:52 (v)
Mozart's Serenade 13 Romance	6:45 (iv)	5:47 (v)
Mozart's Serenade 13 Rondo	3:24 (iv)	2:55 (v)
Sgt Pepper's Lonely hearts	2:01 (i)	2:01 (i)
Something	3:02 (i)	2:59 (i)
Swan Lake theme	3:14 (ii)	3:13 (iii)
Symphony 41 Molto Allegro	8:50 (vi)	8:55 (vi)
With a little help from my friends	2:44 (i)	2:43 (i)
Yellow submarine	2:37 (i)	2:35 (i)

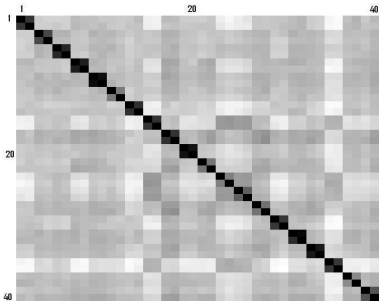

Fig. 5. Confusion Matrix result from using the Longest Common Subsequence distance

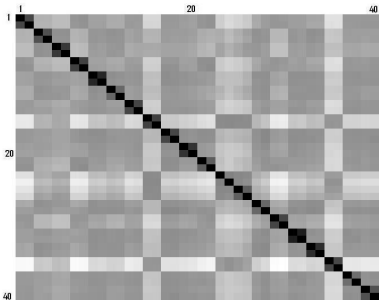

Fig. 6. Confusion Matrix result from using The Levenshtein distance

between the second audio file and every other one, and so on. Since the audio files share a unique prefix if they correspond to the same song, the ideal resulting graphical confusion matrix would be all white with 20 black squares along the main diagonal, each square's wide would have to be of exactly of two columns. Figure 4 corresponds to the experiment using DTW, figure 5 is the graphical confusion matrix when using LCS distance and figure 6 corresponds to the use of the Levenshtein distance.

3 Conclusions

The espectral entropy based string AFP was successfully used in the problem of matching audio performances, the other string AFP of Haitsma-Kalker produces so long strings that for a four minute song a string of 20 690 symbols is obtained while a string of only 320 symbols is obtained with the entropy based AFP, aligning such long strings is impractical and so no tests were included for the Hash string AFP. The three aligning techniques tried, DTW, LCS and Levenshtein distance worked very well, either using the nearest neighbor criterium or simply selecting a threshold every performance matched the other performance of the same song. The flexible string based aligning techniques are more adequate in the issue of monitoring occurrences of performances in audio signals as described in subsection 1.3. We believe that more algorithms developed in the field of matching DNA sequences [5] can be adjusted to their use in Music Information Retrieval using the spectral entropy based string AFP.

References

1. Hu, N., Dannenberg, R.B., Tzanetakis, G.: Polyphonic audio matching and alignment for music retrieval. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (2003)
2. Shalev-Shwartz, S., Dubnov, S., Friedman, N., Singer, Y.: Robust temporal and spectral modeling for query by melody. Proc of ACM SIGIR'02 (2002)

3. Cano, P., Loscos, A., Bonada, J.: Score-performance matching using hmms. Proceedings ICMC99 (1999)
4. Dixon, S.: Live tracking of musical performances using on-line time warping. Proc of the 8th Int Conf on Digital Audio Effects (DAFx'05) (2005)
5. Gusfield, D.: Algorithms on Strings, Trees, and Sequences. Computer Science and Computational Biology. Cambridge University Press (1997)
6. Navarro, G., Raffinot, M.: Flexible Pattern Matching in Strings. Practical On-Line Search for Texts and Biological Sequences. Volume 17. Cambridge University Press (2002)
7. Ibarrola, A.C., Chavez, E.: A very robust audio-fingerprint based on the information content analysis. IEEE transactions on Multimedia (submitted) available: http://lc.fie.umich,mx/∼camarena.
8. Hellmuth, O., Allamanche, E., Cremer, M., Kastner, T., NeuBauer, C., Schmidt, S., Siebenhaar, F.: Content-based broadcast monitoring using mpeg-7 audio fingerprints. International Symposium on Music Information Retrieval ISMIR (2001)
9. Haitsma, J., Kalker, T.: A highly robust audio fingerprinting system. IRCAM (2002)
10. P. Cano, E.B., Kalker, T., Haitsma, J.: A review of algorithms for audio fingerprinting. IEEE Workshop on Multimedia Signal Processing (2002) 169–167
11. Shannon, C., Weaver, W.: The Mathematical Theory of Communication. University of Illinois Press (1949)
12. Shen, J.L., Hung, J.w., Lee, L.s.: Robust entropy-based endpoint detection for speech recognition in noisy environments. In: Proc. International Conference on Spoken Language Processing. (1998)
13. You, H., Zhu, Q., Alwan, A.: Entropy-based variable frame rate analysis of speech signal and its applications to asr. In: Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). (2004)
14. Ibarrola, A.C., Chavez, E.: A robust, entropy-based audio-fingerprint. IEEE International Conference on Multimedia and Expo 2006 (ICME2006) To appear (2006)
15. Group, M.A.: Text of ISO/IEC Final Draft International Standar 15938-4 Information Technology - Multimedia Content Description Interface - Part 4: Audio. MPEG-7 (2001)
16. Martin, R.: Noise power spectral density estimation based on optimal smoothing and minimum statistics. IEEE Transactions on Speech and Audio Processing **9** (2001) 504–512
17. Sakoe, H., Chiba, S.: Dynamic programming algortihm optimization for spoken word recognition. IEEE transactions on Acoustics and Speech Signal Processing (ASSP) (1978) 43–49