Representing Audio Data by FS-Trees and Adaptable TV-Trees

Alicja A. Wieczorkowska[1], Zbigniew W. Raś[2,3], and Li-Shiang Tsay[2]

[1] Polish-Japanese Institute of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland
[2] University of North Carolina, Department of Computer Science, Charlotte, N.C. 28223, USA
[3] Polish Academy of Sciences, Institute of Computer Science, Ordona 21, 01-237 Warsaw, Poland
{awieczor,ras,ltsay}@uncc.edu

Abstract. An automatic content extraction from multimedia files based both on manual and automatic indexing is extensively explored. However, in the domain of musical data, an automatic content description of musical sounds has not been broadly investigated yet and still needs an intensive research. In this paper, spectro-temporal sound representation is used for the purpose of automatic musical instrument recognition. Assuming that musical instruments can be learned in terms of a group of features and also based on them either automatic or manual indexing of an audio file is done, Frame Segment Trees (FS-trees) can be used to identify segments of an audio marked by the same indexes. Telescopic vector trees (TV-trees) are known from their applications in text processing and recently in data clustering algorithms. In this paper, we use them jointly with FS-trees to construct a new Query Answering System (QAS) for audio data. Audio segments are returned by QAS as answers to user queries. Heuristic strategy to build adaptable TV-trees is proposed.

1 Introduction

Indexing and searching multimedia data is a non-trivial task, which requires signal analysis and processing. Standardization of content labeling of multimedia data is addressed by Moving Picture Experts Group in MPEG-7 standard, named Multimedia Content Description Interface [11]. This standard provides numerous descriptors for labeling audio and visual data. However, algorithms for searching through the labeled data are behind the scope of MPEG-7, since their standardization is not required, thus leaving space for further development. Therefore, research on efficient searching through audio data represented by descriptors is still desirable. Additionally, the use of new descriptors may be necessary for more efficient searching or more detailed labeling of multimedia data. This is why we decided to explore a new data structure based on TV-trees jointly with FS-trees for representation of multi-dimensional audio data, in order to allow efficient searching of specific audio segments (fulfilling not only index

type properties, but also some similarity criteria). We focus our attention on musical instrument sounds.

2 Descriptors of Musical Audio Data

In the research published so far, various methods for musical data parameterization have been used. The parameters (descriptors) can be based on time domain features, spectrum, or/and other analyses. Time-domain parameters of musical instrument sounds may describe the duration of particular phases of the sound (the attack, quasi-steady state and ending transient) or properties of sound wave, like number of zero-crossings in time unit etc. [14]. Also, features of sound envelope can be used as descriptors [21].

Descriptive features of sound can also be extracted from spectral analysis. These features may describe contents of the selected groups of harmonics in spectrum, like even/odd harmonics, or lower/middle/higher harmonics [14], [19]. Also fundamental frequency and its variance (vibrato) are used as descriptors [16], [24]. Other parameters describe statistical properties of spectrum, like average amplitude and frequency deviations, average spectrum, standard deviation, auto- and cross-correlation functions [1], [5]. Such properties of spectrum as higher order moments, such as skewness and kurtosis, as well as brightness and spectral irregularity, are also used for musical sound parameterization [8], [23].

Sound descriptors can be also derived from other analyzes. Results published so far are based on constant-Q coefficients [4], [12], auditory model [16], cepstral and mel-cepstrum coefficients and derivatives [2], [4], [7], multidimensional scaling analysis trajectories [12], wavelets [13], [23], Bark scale bands [7] and other [9], [25]. A set of descriptors for audio content description is also provided in MPEG-7 standard [11]; these descriptors are aimed to aid musical instrument timbre description, audio signature, sound description etc. Apart from higher-level descriptors, the audio framework in MPEG-7 includes 17 temporal and spectral low-level descriptors, divided into the following groups [15]:

- basic: instantaneous waveform and power values,
- basic spectral: log-frequency power spectrum envelopes, spectral centroid, spectrum spread, and spectrum flatness,
- signal parameters: fundamental frequency and harmonicity of signal,
- timbral temporal: log attack time and temporal centroid,
- timbral spectral: spectral centroid, harmonic spectral centroid, and harmonic spectral deviation, harmonic spectral spread, harmonic spectral variation,
- spectral basis representations: spectrum basis and spectrum projection.

These features can be used to describe a segment with a summary value or with a series of sampled values that apply to the entire segment, with exception of the timbral temporal group, applied to segments as a whole only.

Sound description schemes presented above focus on digital and compact sound representation for classification purposes. Assuming feature vector is given, our aim is to propose a new searching method through audio data based on tree-like structures.

In our research, we are dealing with recordings containing singular sounds of musical instruments. We apply descriptors which have been already used in experiments and which gave good results, i.e. up to 70% of correct classification, which is comparable with results of similar worldwide research [9], [21]. These descriptors characterize temporal and spectral features of short sound frames and sound as a whole. Temporal group includes the following descriptors [21]:

- *Length*: Signal length,
- *Attack*, *Steady* and *Decay*: Relative length of the attack, i.e. the onset portion of sound (till reaching 75% of maximal amplitude), quasi-steady state (after the end of attack, till the final fall under 75% of maximal amplitude) and decay time (the rest of the signal), respectively,
- $Maximum$: Moment of reaching maximal amplitude,
- $Env1, \ldots, Env7$: Envelope parameters: average values of envelope of sound amplitude in consequent 7 intervals of equal width,
- $EnvFill$: Area of envelope,
- *Cluster*: Number of cluster (obtained through clusterization of envelope vectors for the whole data set into 6 clusters) closest to the analyzed sound, with respect to Euclidean metric.

Spectral descriptors include [23]:

- $Frequency$: Fundamental frequency,
- Br and Ir: Brightness Br and irregularity Ir of spectrum, defined as

$$Br = \frac{\sum_{n=1}^{N} n\, A_n}{\sum_{n=1}^{N} A_n} \quad Ir = \log\left(20 \sum_{k=2}^{N-1} \left| \log \frac{A_k}{\sqrt[3]{A_{k-1} A_k A_{k+1}}} \right| \right) \tag{1}$$

  where A_n - amplitude of n^{th} harmonic, N - number of available harmonics,
- Ev and Od: Contents of even Ev and odd Od harmonics in spectrum:

$$Ev = \sqrt{\frac{\sum_{k=1}^{M} A_{2k}^2}{\sum_{n=1}^{N} A_n^2}} \qquad Od = \sqrt{\frac{\sum_{k=2}^{L} A_{2k-1}^2}{\sum_{n=1}^{N} A_n^2}} \tag{2}$$

  where $M = \lfloor N/2 \rfloor$ and $L = \lfloor N/2 + 1 \rfloor$,
- $Tristimulus1, 2, 3$: Tristimulus [19] parameters given by:

$$Tr_1 = \frac{A_1^2}{\sum_{n=1}^{N} A_n^2} \quad Tr_2 = \frac{\sum_{n=2,3,4} A_n^2}{\sum_{n=1}^{N} A_n^2} \quad Tr_3 = \frac{\sum_{n=5}^{N} A_n^2}{\sum_{n=1}^{N} A_n^2} \tag{3}$$

All these parameters were calculated for the whole sounds of musical instruments of definite pitch. Additionally, spectral parameters and maximal amplitude were calculated for consequent short frames, covering this sound. The length of analyzing frame was equal four times the sound period. Most common length of analyzing frame is about 20–30 ms [2], [4], [5], [7], whereas in our case it is 2–125 ms. Shorter frames allow observation of fast changes, especially for pizzicato sounds and during transients; longer frames are taken in case of the lowest sounds. Calculation of sound descriptors using sliding window allows observation of changes of parameters values in time. The obtained attribute values produce time series, which are next analyzed to discover patterns characterizing sound evolution in time, specific for some groups of sounds [21]. This way, new descriptors of musical sounds are generated.

Observations of parameters using sliding window is also a step towards experiments with ordinary, unstructured recordings, which require segmentation through discovering changes in sound characteristics.

3 Classification of Data

The descriptors presented in the previous section were applied to recordings of consequent singular sounds of musical instruments of definite pitch, taken from MUMS stereo recordings [18]. These CD's are widely used in musical instrument sound research [7], [8], [12], [16], [24], so they can be considered as a standard. The parameterized 667 sounds represent 11 instruments, played using various articulation techniques. The data objects are grouped into the following 19 classes, with respect to both instrument and articulation [24]: violin vibrato, violin pizzicato, viola vibrato, viola pizzicato, cello vibrato, cello pizzicato, double bass vibrato, double bass vibrato, double bass pizzicato, flute, oboe, b-flat clarinet, trumpet, trumpet muted, trombone, trombone muted, French horn, French horn muted, and tuba. These objects are stored in a decision table. Formally, the decision table is defined as $\mathbf{A} = (U, A \cup \{d\})$, where each attribute $a \in A \cup \{d\}$ is identified with a function $a : U \rightarrow V_a$ from the universe of objects U into the set V_a of all possible values of a. Values $v_d \in V_d$ correspond to mutually disjoint decision classes of objects. In our case, each element $u \in U$ corresponds to a sound sample, elements $a \in A$ are numeric features corresponding to sound descriptors and the decision attribute $d \notin A$ labels particular object (sound) with integer code representing the instrument.

The data placed in decision table can be used for generalization, i.e. prediction of decision class based on information provided by some attributes of new objects. This is achieved through construction of classifiers, such as decision trees, rule-based classifiers, k-nearest neighbor classifiers, neural nets, etc. The issue of musical instrument sound recognition has been brought up in several research studies, applying various methods [10], [21]. The most common one is k-nearest neighbor algorithm, applied in [7], [8], [12], [16]. Brown in her research [4] applied clustering and Bayes decision rules, using k-means algorithm to calculate clusters, and forming Gaussian probability density functions from

the mean and variance of each of the clusters. Martin and Kim (1998) used maximum a posteriori classifiers, based on Gaussian models obtained through Fisher multiple-discriminant analysis. Gaussian classifier was also used by Eronen and Klapuri (2000). Other applied methods include binary trees, used in [23], hidden Markov models, reported in [2], and rough set approach [24].

4 Audio Content Representation using FS-trees and Adaptable TV-trees

Rules describing different sounds in terms of features from A can be extracted from the table $\mathbf{A} = (U, A \cup \{d\})$. When applied to a new audio signal they can be considered as a tool for its automatic indexing. For audio files containing only sounds of singular instruments this tool has been already successful [23] due to the high confidence of rules describing singular sounds. Now, assuming that indexing of audio files both automatic and manual is done, FS-trees [22] can be used for a compact content description representation of audio data and also for efficient processing of audio queries based on this indexing. But, users may want to submit additional queries which are audio signals represented as frames. Query Answering System (QAS), based not only on FS-trees but also on trees similar to binary Telescopic Vector trees (TV-trees) [22], [25] is used by us to search for frames similar to a frame representing user's query.

Now, we outline the strategy for construction our TV-trees. Each audio signal is divided into frames of length four times the fundamental period of the sound. Each frame is represented as a vector consisting of K acoustic descriptors. So, any collection of audio signals can be seen as a set of K-dimensional vectors. We represent this set as $(K \times N)$-matrix where N is the number of frames in all audio signals in the database. If needed, we reduce K to a smaller number using Singular Value Decomposition method [22]. The notion of a distance between two vectors, used by us, is a standard one. These vectors are seen as points in K-dimensional space. They are partitioned into disjoint clusters in a such way that points similar with respect to maximal number of coordinates are kept in each cluster. These coordinates are called active dimensions. To define a as an active dimension in a cluster, we require that the span of values of a in that cluster has to be below some threshold value. For example, if $\{1, \ldots, 100\}$ is the domain of an attribute and its corresponding threshold value is $1/20$, then this attribute is active in a cluster of N vectors if the distance between values of this attribute for any 2 vectors in that cluster is not greater than 5. Assume now that our goal is to represent the set of N points as a TV-tree of order 2, which means that only 2 clusters per node can be constructed. So, we divide the set of N points into 2 clusters maximizing the total number of active dimensions in both clusters. For each cluster we repeat the same procedure, again maximizing the total number of active dimensions in the corresponding sub-clusters. For instance, if $\{[5,3,20,1,5], [0,0,18,41,4], [0,0,19,39,5], [11,10,2,0,5]\}$ is the initial cluster, then the following 2 sub-clusters are generated: $\{[0,0,18,41,4], [0,0,19,39,5]\}$, $\{[5,3,20,1,5], [11,10,2,0,5]\}$. The initial cluster has only the

last dimension active. After split, the first sub-cluster has 5 dimensions active and second one has the last 2 dimensions active. We continue this procedure till all sub-clusters are relatively dense (all points are close to each other with respect to all dimensions). For instance, in the example above, the first sub-cluster is dense. The underlying structure for this method is a binary tree with nodes storing information about the center (c) of a corresponding cluster, the smallest radius (r) of a sphere containing this cluster, and its list of active dimensions [25].

Our heuristic procedure to construct a binary TV-tree (part of QAS) for a collection of audio signals is similar to the one used in Rosetta or See5 system for discretizing numerical attributes [20], [17]. For m-element domain of an attribute, $m-1$ splitting points are considered (alternatively, the splitting points can be placed between consecutive dense groups of values of an attribute). Saying more precisely, if v_1, v_2 are neighboring values of the attribute a, then an audio signal with value of a less than or equal to $[v_1 + v_2]/2$ is placed in the left subtree and all other audio signals are placed in the right subtree. When this is done, the total number of active dimensions in both left and right subtree is checked. We repeat this step for each attribute and for its all splitting points mentioned above. A split which gives the maximal number of active dimensions, for both left and right sub-tree, is the winning split and it is used to build two children of the current node. The above procedure is recursively repeated at two children nodes just created.

Now, let us assume that QAS is based both on a TV-tree and an FS-tree and user submitting an audio query to QAS is looking for audio files satisfying certain properties corresponding to indexes used in FS-trees. User's audio query contains also a sub-query represented in QAS as a vector similar to vectors stored in a TV-tree. For this type of queries, FS-tree is searched first for audio files satisfying desired properties (index-based search). Next, the TV-tree is searched to identify which files retrieved from FS-tree also satisfy the additional sub-query. Our initial results with respect to precision and recall from testing QAS are very encouraging.

5 Construction of Adaptable TV-trees

Let us assume that audio data are stored in a relational table, where attributes are either features computed from the raw data given by users or indexes provided by experts. By an adaptable TV-tree, we mean a TV-tree built from a relational table, where each splitting attribute is chosen not only on the basis of how to maximize the total number of active features in children nodes it generates, but also on the basis which attributes are used most frequently in user queries. Such attributes are called *popular*. The idea here is to make popular attributes active as soon as we can so they are used by QAS in searching its TV-tree at the very beginning of a query answering process.

Assume that a_1, a_2, a_3,...,a_K is an ordered set (with respect popularity) of all acoustic descriptors in the relational table (the most popular descriptors

are listed first). Let k_i is a frequency of occurrence of a descriptor a_i in vectors representing audio queries submitted to QAS. In order to build an adaptable TV-tree we consider a function from $\{a_1, a_2, a_3, ..., a_K\}$ into $\{1, 2, 3, ..., K\}$, defined as $f(a_i) = (K - i + 1)$. To evaluate a relational table from the point of view of a descriptor a_i, we multiply $1/f(a_i)$ by the number of cuts needed in $Dom(a_i)$ to make a_i active. Let's denote the resulting number by $h(a_i)$. Descriptor with a cut making $h(a_i)$ the smallest is chosen as the new node of adaptable TV-tree. This tree is called adaptable since the construction of its nodes is driven by a function $h(a_i)$, which may change in time as new queries are submitted to QAS. For simplicity reason only one of the easiest versions of the function $f(a_i)$ was presented here, but more complex definitions of $f(a_i)$ have been tested as well.

6 Conclusion

Automatic indexing of multimedia databases is of great importance, and ISO/IEC decided to provide MPEG-7 standard for multimedia content description. However, this standard does not comprise the extraction of descriptors (nor search algorithms). Therefore, there is a need to elaborate extraction of sound descriptors that would be attached to sound files.

This paper reviews some audio descriptors based on various kinds of sound analysis. We use them, for instance, to learn definitions of musical instruments (to construct a new audio descriptor). This new audio descriptor is only an example of a descriptor useful for an automatic indexing of musical files. Finally, we use all our audio descriptors to construct QAS based on FS-tree and adaptable TV-tree to allow efficient access to audio data. The initial results related to precision and recall of our QAS are good.

The main goal of this paper is to find a new data representation suitable for handling audio queries based not only on indexes, but also containing semantic type of information, which cannot be expressed in index-type languages.

Acknowledgements

This research was sponsored by the Research Center of PJIIT, supported by the Polish National Committee for Scientific Research (KBN).

References

1. Ando, S., Yamaguchi, K.: Statistical Study of Spectral Parameters in Musical Instrument Tones. *J. Acoust. Soc. of America* 94(1) (1993) 37–45
2. Batlle, E., Cano, P.: Automatic Segmentation for Music Classification using Competitive Hidden Markov Models. *Int. Sym. Mus. Inf. Retr.* Plymouth, MA (2000)
3. Beauchamp, J. W., Maher, R., Brown, R. . Detection of Musical Pitch from Recorded Solo Performances. 94th AES Convention, preprint 3541, Berlin (1993)
4. Brown, J.: Computer identification of musical instruments using pattern recognition with cepstral coefficients as features. *J.Acoust.Soc.Am.* 105 (1999) 1933–1941

5. Brown, J., Houix, O., McAdams, S.: Feature dependence in the automatic identification of musical woodwind instruments. *J. Acoust. Soc. Am.* 109 (2001) 1064–1072
6. Cook, P. R., Morrill, D., Smith, J. O.: An Automatic Pitch Detection and MIDI Control System for Brass Instruments. Invited for special session on Automatic Pitch Detection, Acoustical Society of America, New Orleans (1992)
7. Eronen, A., Klapuri, A.: Musical Instrument Recognition Using Cepstral Coefficients and Temporal Features. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2000* Plymouth, MA. (2000) 753–756
8. Fujinaga, I., McMillan, K.: Realtime recognition of orchestral instruments. *Proceedings of the International Computer Music Conference* (2000) 141–143
9. Herrera, P., Amatriain, X., Batlle, E., Serra X.: Towards instrument segmentation for music content description: a critical review of instrument classification techniques. In *Proc.Int.Sym.Music Inf.Retr. (ISMIR 2000)*, Plymouth, MA (2000)
10. Herrera, P., Peeters, G., Dubnov, S.: Automatic Classification of Musical Instrument Sounds *Journal of New Music Research* 32(1) (2003)
11. ISO/IEC JTC1/SC29/WG11: MPEG-7 Overview (2002)
12. Kaminskyj, I.: Multi-feature Musical Instrument Classifier. *MikroPolyphonie* 6 (2000) (online journal at http://farben.latrobe.edu.au/)
13. Kostek, B. and Czyzewski, A.: Representing Musical Instrument Sounds for Their Automatic Classification. *J. Audio Eng. Soc.* 49(9) (2001) 768–785
14. Kostek, B., Wieczorkowska, A.: Parametric Representation of Musical Sounds. *Arch. Acoustics* 22(1), Inst. Fund. Tech. Research, Warsaw, Poland (1997) 3–26
15. Lindsay, A. T., Herre, J.: MPEG-7 and MPEG-7 Audio – An Overview. *J. Audio Eng. Soc.* 49(7/8) (2001) 589–594
16. Martin, K., Kim, Y.: 2pMU9. Musical instrument identification: A pattern-recognition approach. 136-th meeting Acoustical Soc. America, Norfolk, VA (1998)
17. Øhrn, A., Komorowski, J., Skowron, A., Synak, P. The design and implementation of a knowledge discovery toolkit based on rough sets: The ROSETTA system. In: Polkowski, L., Skowron, A. (Eds.), *Rough Sets in Knowledge Discovery 1: Methodology and Applications*, number 18 in Studies in Fuzziness and Soft Computing, chapter 19, Physica-Verlag, Heidelberg, Germany (1998) 376–399
18. Opolko, F., Wapnick, J.: MUMS – McGill University Master Samples. CD's (1987)
19. Pollard, H. F., Jansson, E. V.: A Tristimulus Method for the Specification of Musical Timbre. *Acustica* 51 (1982) 162–171
20. Quinlan, J. R.: *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, California (1993)
21. Ślęzak, D., Synak, P., Wieczorkowska, A., Wróblewski, J.: KDD-based approach to musical instrument sound recognition. In: Hacid, M.-S., Ras, Z., Zighed, D., Kodratoff, Y. (Eds.), *Found. Intel. Syst.* LNCS/LNAI 2366, Springer (2002) 29–37
22. Subrahmanian, V. S.: *Multimedia Database Systems.* Morgan Kaufmann Publishers, San Francisco, CA (1998)
23. Wieczorkowska, A. A.: The recognition efficiency of musical instrument sounds depending on parameterization and type of a classifier. PhD thesis (in Polish), Technical University of Gdansk, Poland (1999)
24. Wieczorkowska, A.: Rough Sets as a Tool for Audio Signal Classification. In: Ras, Z. W., Skowron, A. (Eds.), *Foundations of Intelligent Systems* (pp. 367–375). LNCS/LNAI 1609, Springer (1999)
25. Wieczorkowska, A. A., Raś, Z. W. Audio Content Description in Sound Databases. In: Zhong, N., Yao, Y., Liu, J., Ohsuga, S. (Eds.), *Web Intelligence: Research and Development* LNCS/LNAI 2198, Springer (2001) 175–183