AUTOMATIC EXTRACTION OF MUSIC DESCRIPTORS FROM ACOUSTIC SIGNALS

Pachet François, Zils Aymeric
Sony CSL Paris
{pachet, zils}@csl.sony.fr

ABSTRACT

High-Level music descriptors are key ingredients for music information retrieval systems. Although there is a long tradition in extracting information from acoustic signals, the field of music information extraction is largely heuristic in nature. We present here a heuristic-based generic approach for extracting automatically high-level music descriptors from acoustic signals. This approach is based on Genetic Programming, used to build relevant features as functions of mathematical and signal processing operators. The search of relevant features is guided by specialized heuristics that embody knowledge about the signal processing functions built by the system. Signal processing patterns are used in order to control the general processing methods. In addition, rewriting rules are introduced to simplify overly complex expressions, and a caching system further reduces the computing cost of each cycle. Finally, the features build by the system are combined into an optimized machine learning descriptor model, and an executable program is generated to compute the model on any audio signal. In this paper, we describe the overall system and compare its results against traditional approaches in musical feature extraction à la Mpeg7.

1. INTRODUCTION

The exploding field of Music Information Retrieval has recently created extra pressure to the community of audio signal processing, for extracting automatically high level music descriptors. Indeed, current systems propose users i.e. the possibility to access music titles based on their actual content, rather than on file names. Existing systems today are mostly based on editorial information (e.g. Kazaa), or metadata which is with millions of music titles (e.g. the peer-to-peer systems such as Kazaa) and query functions limited usually to string matching on title names. The natural extension of these systems is content-based access, entered manually, either by pools of experts (e.g. All Music Guide) or in a collaborative manner (e.g. MoodLogic). Because these methods are costly and do not allow scale up, the issue of extracting automatically high-level features from acoustic signals is key to the success of online music access systems.

Extracting automatically content from music titles is a long story. Many attempts have been made to identify dimensions of music that are perceptually relevant and can be extracted automatically. One of the most known is tempo or beat. Beat is a very important dimension of music that makes sense to any listener. [1] introduced a beat tracking system that successfully computes the beat of music signals with good accuracy. There are, however, many other dimensions of music that are perceptually relevant, and that could be extracted from the signal: the presence of voice in a music title, the perceived intensity (subjective impression of energy that music titles convey: with the same volume, a Hard-rock music title conveys more energy than an acoustic guitar ballad with a soft voice), difference between “live” and studio recording, recognition of typical musical genres, evaluation of the danceability of a song, etc. Yet this information is difficult to extract automatically, because music signals are usually highly complex, polyphonic in nature, and incorporate characteristics that are still poorly understood and modeled.

2. TOWARDS AUTOMATIC EXTRACTION OF MUSICAL DESCRIPTORS

2.1. The traditional method: combination of generic Low-Level Descriptors

Typically, the design of a descriptor extractor consists in combining generic Low-Level Descriptors (LLDs) as relevant characteristics of acoustic signals (features), using machine learning algorithms (see, e.g. [2], [3], [4]):

- Several features are computed. A typical reference for audio signal features is the Mpeg7 standardization process [5] that proposes a battery of LLDs for describing basic characteristics of audio signals.
- the most relevant features are selected and combined into machine learning processes, to provide an optimal model for the descriptor.

The traditional method sketched above works well only for relatively easy problems; problems for which generic low level features are adapted. However, generic features can only extract information which is “predominant” in the signal, and are, by definition, unable to focus on specific, problem-dependent properties. The core assumption of this paper is precisely that in order to solve more difficult problems one needs specific features adapted to the problem at hand.

Our second assumption is that these specific features can be extracted automatically as compositions of signal processing operations.

2.2. Improving generic LLD combination using automatic operators composition

The design of specific features that are relevant for a given description problem is usually done by hand by signal processing experts. This section introduces the idea of generating automatically such specific features adapted to a particular problem.

Although there is no known general paradigm for designing domain-specific features, their design usually follows some sort of patterns. One of them consists in filtering the signal, splitting it into frames, applying specific treatments to each segment, then aggregating all these results back to produce a single value. This is typically the case of the beat tracking system described in [1] that yields eventually a float representing the tempo. The same applies to timbre descriptors proposed in the music information retrieval and more generally to most audio descriptors described in the literature ([6]). Of course, this global scheme of expansion/reduction is under specified, and an infinite number of such schemes could be envisaged.

Our goal is therefore to design a system that is able to 1) search automatically relevant signal processing features, seen as compositions of functions and build a model of the descriptor and 2) reduce the search space significantly using generic knowledge on signal processing operators.

3. EDS, AN EXTRACTOR DISCOVERY SYSTEM

We describe here the main ingredients of the EDS system: the definition of a description problem, the automatic construction of relevant signal processing functions, and their combination into a general descriptor model.

3.1. Definition of a description problem

The definition of the description problems handled by the system has to remain simple to preserve the generality of the approach. One simple way to define a description problem is to use the supervised learning approach: a set of labeled signals, also called *learning database*, defines the description problem. These labels are either numeric values, such as an evaluation of their “musical energy” (between 0 and 1), or a class label, such as the “presence of a singing voice or not”, or the genre chosen in a given taxonomy. The system will then finds the rules of the labeling of the signals, i.e. the model of the descriptor, by designing a function which produces outputs as close as possible to the learning database labels.

3.2. General principles of EDS

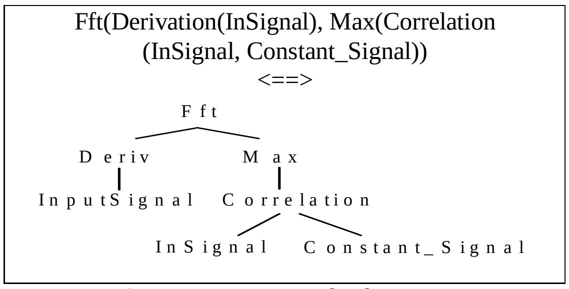
Our system EDS composes automatically operators to discover features as signal processing functions that are optimal for a given descriptor extraction task. Its global architecture consists in two fully automatic parts: modeling of the descriptor and synthesis of the extractor.

The modeling of the descriptor is the main part of EDS. It consists in searching automatically for relevant features and then for the optimal model that combines these features. The search for specific features is based on genetic programming, a well-known technique for exploring search spaces of function compositions (see [7]). The genetic programming engine composes automatically signal processing operators to build arbitrarily complex functions, and evaluates their relevance to extract a given descriptor on a given learning database. The evaluation of a function is very costly, as it involves complex signal processing on whole audio databases. Therefore, to limit the search, a set of heuristics are introduced to improve the a priori relevance of the created functions, as well as rewriting rules to simplify functions before their evaluation. Once the system has found relevant features, it combines them to feed them into various machine learning models, and then optimizes the model parameters.

The synthesis part consists in generating an executable file to compute the best model on any audio signal. This program allows computing this model on arbitrary audio signals, to predict their value for the modeled descriptor.

3.3. Automatic construction of features

Functions are represented in EDS as compositions of basic operations applied on an arbitrary input audio signal. The basic operators can be mathematical, such as taking the mean values of a set, or can process a signal, temporally (such as correlation), or spectrally (such as a low-pass filtering). In addition, some operations are parameterized using constant values (like cut-off frequencies), or external signals (for example a correlation with a fixed reference signal). Consequently, the functions can be represented as signal processing operators trees (see Fig.1):


Fft(Derivation(InSignal), Max(Correlation (InSignal, Constant_Signal))

<==>

Fig. 1: The syntactic tree of a function in EDS

3.4. Automatic construction of features

The automatic construction of correct functions relies on the control of the types of data handled by the functions, and on the introduction of signal processing expertise as heuristics.

3.4.1. Data Types

The need for typing is well-known in Genetic Programming, to ensure that the functions generated are at least syntactically correct. Different type systems have been proposed for GP, such as strong typing ([8]) that mainly differentiate between the “programming” types of the inputs and outputs of functions. To control the physical processes in EDS, we need to distinguish how the functions built by the system handle the data, at the level of their “physical dimension”. For instance, audio signals and spectrum are both as vectors of floats, but are different in their

dimensions: a signal is a time to amplitude representation, a spectrum frequency to amplitude.
Our typing system, based on the following constructs, represents this difference, to ensure that our resulting functions make sense. Using only three physical dimensions (time "t", frequency "f", and amplitudes or non-dimensional data "a"), we are able to represent most of the data types handled by the system, by building atomic (single value), vector (multiple values), and functional (data of a given type evolving in a dimension of another type) types.

3.4.2. Operator types

The operations in EDS transform physically the data, and can therefore be specified using the typing system. For each operator, we define typing rules that provide the type of its output data, depending on the types of its input data. The typing rules are usually reduced into a dimensionality rule and a transformation rule. For example, the "Spectrum" operation transforms a signal of type "t:a" into a frequency spectrum of type "f:a".

3.4.3. Controlling processing methods using patterns

The types of data handled by a function are a signature of the general processing methods used in the function. In order to control globally the processing methods through the successive types of data handled by the functions, we have introduced "generic operators" that stand for one or several random real operator(s) whose output types are forced. EDS can deal with three different generic operators (notated "*", "!", and "?") that have different functionalities. These generic operators allow specifying locally the processes to use in a function. By composing them in *patterns*, we describe a global set of processes to apply on an audio signal to obtain a final value. For instance, the simple *pattern* "?_a (!_Va (Split (*_t:a (Signal))))" is a translation of the general extraction scheme presented in 3.1, standing for the process "Temporal domain – Split – Agregation". There are various ways to instantiate this pattern, for example "Sum$_{a}$ (Square$_{Va}$ (Mean$_{Va}$ (Split$_{Vt:a}$ (HpFilter$_{t:a}$ (Signal$_{t:a}$, 1000Hz)))))".
Patterns are specified in the EDS algorithm in order to guide the search of functions.

3.4.4. Heuristics

In order to guide the instantiation of the patterns, we need to introduce knowledge in the system, as signal processing heuristics that represent the know-how of signal processing experts, about functions seen a priori, both favoring a priori interesting functions and ruling out obviously non-interesting ones.
A heuristic in EDS associates a score to a potential composition of operators, between 0 (forbidden composition) and 10 (very recommended composition). These scores are used when EDS builds a new function, to select the candidates between all the possible operations.
Basically, the heuristics allow controlling the structure of the functions, avoiding bad combination of operations, ranging constant parameters values, and avoiding usually useless operations.

3.4.5. Automatic construction of functions

The automatic synthesis of functions is performed in bottom-up fashion, starting from the input signal, and grafting sequentially the operators one after the other up to the top of the tree, all the generic operators being instantiated, i.e. replaced by real operators.

3.5. Search for optimized features

The function search part in EDS consists in building signal processing functions that are increasingly relevant, using an algorithm based on genetic programming, i.e. the application of genetic search to the world of functions, as introduced by [9]. The algorithm builds a population of functions from a given pattern, and tries to improve them iteratively by applying various genetic transformations on them. Running this algorithm once provides one optimal function to be used in the final model. Therefore, this algorithm is run N times to build N optimized functions constituting the final feature set used in the final model of the descriptor.

During the genetic search, each new population is created by applying genetic transformations on the most relevant functions of the current population. 3 main transformations are used in EDS: variations of the constant parameters (keeps the tree structure a function and applies variations on its parameters such as cut-off frequencies or window sizes), mutation of operations in the function, and crossover of two parts of two functions.

Eventually, in order to search more efficiently, rewriting rules (simplifying functions using a fixed point mechanism) and a caching mechanism (keeping the most useful results in memory, depending on their computation time, utility, and size) have been included in the system.

3.6. Final model of the descriptor

After running the genetic search, EDS finds relevant features well adapted to the description problem at hand. These features are then combined into an optimized model of the descriptor, using generic machine learning techniques (kNN, Neural Nets, Decision Trees, etc…).

Each of these models carries with it a certain number of parameters such as the number of neighbours in the k-NN method, or the number of layers for the Neural Networks. The processes of selecting the optimal model with the optimal parameters are entirely automated in EDS.
The final descriptor model is the best model found, defined by a set of relevant features and a modelling technique with optimized parameters, such as for instance:
"6-NN ('Max(Fft(S))','Variance(Autocorrelation(S))')".

The performance of this model is evaluated on a test database (different from the learning database) for assessing definitively its performance.
And finally, a self-executable extractor is generated automatically to compute the model on a .wav signal.

4. PERFORMANCE OF THE SYSTEM

We present here the performance on the two steps of EDS: the relevance of the features and the quality of the models found by EDS, compared to those made using the Mpeg7 LLDs dataset (called "LLDs").

4.1. Regression problem: Musical energy

The problem consists in providing a model of the subjective energy of musical extracts, based on the results of perceptive tests (see [10]). This descriptor addresses the intuitive difference there is, for example, between a punchy punk-rock song with loud saturated guitars and screaming voice conveys and an acoustic guitar ballad with a soft voice, at a constant volume level.

4.1.1. LLDs

The best LLD found was "Mean (SpectralSkewness (Split (Signal, 250.0)))", with correlation=0,548 on learn and 0,658 on test. A forward features selection (see[12]) on the LLDs kept 25 features to build the final model of musical energy. The best method found was a Model Tree provided a correlation=0.698 (0.810 on test), which corresponds to an average model error of 12.80% (13,26%).

4.1.2. EDS

The best function found by EDS is "BestEDS(Signal)=Square (Log10 (Mean (Min (Fft (Split (Testwav, 4009))))))", with correlation=0,744 on learn, and 0,812 on test. A forward selection kept 4 features to build the final model of musical energy. The best method found was a Linear Regression that provided a correlation=0.780 on learn, and 0.836 on test, which corresponds to an average model error of 11.52% (13,06%).

4.2. Objective classification problem: Presence of singing voice

The problem consists in providing a model that allows detecting the presence of singing voice in polyphonic audio signals (see [11]).

4.2.1. LLDs

The best LLD found was "SpectralSpread (Testwav)" (Fisher = 0,282 on learn, and 0,215 on test). A forward selection kept 8 features. The best method found is a Naïve Bayes classifier providing a 72% of good classification on learn and 69.5% on test.

4.2.2. EDS

The best function found by EDS is "Log10 (Range (Derivation (Sqrt (Blackman (MelBands (Testwav, 24.0))))))" (Fisher=1,209 on learn, and 0,831 on test).
A forward selection kept 12 features to build the final EDS model of musical energy. The best method found was a kNN classifier providing 86.5% of good classifications on learn, and only 78.5% on test.

5. CONCLUSION

We have introduced a new approach for designing automatically efficient extractors for high-level audio descriptors. Although it uses a limited palette of signal processing functions, the proposed system, EDS, already produces better results than standard approaches using the Mpeg7 generic features.
The generality of the approach allows EDS to address the whole class of extraction problems in the large, from the detection of "live" recordings or the modeling of music danceability or percussivity, etc...
Substantial increase in performance is expected by extending the palette of signal operators to more refined operators, as well as in adding more refined heuristics and rewriting rules to prune the search space.

6. REFERENCES

[1] Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. J. Acoust. Soc. Am. (JASA) 103:1 (Jan 1998), pp 588-601.
[2] Eric D. Scheirer, and Malcolm Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. Proc. ICASSP '97.
[3] P. Herrera, A. Yeterian, F. Gouyon. Automatic classification of drum sounds: a comparison of feature selection methods and classification techniques. Proceedings of 2nd International Conference on Music and Artificial Intelligence, Edinburgh, Scotland, 2002.
[4] Geoffroy Peeters, Xavier Rodet. Automatically selecting signal descriptors for sound classification. Proceedings of the 2002 ICMC, Goteborg (Sweden), September 2002.
[5] Perfecto Herrera, Xavier Serra, Geoffroy Peeters. Audio descriptors and descriptors schemes in the context of MPEG-7. Proceedings of the 1999 ICMC, Beijing, China, October 1999.
[6] JJ Aucouturier, François Pachet. Music similarity measures: what's the use ? In proceedings of the 3rd international symposium on music information retrieval (ISMIR02), Paris, October 2002.
[7] John R. Koza. Genetic Programming: on the programming of computers by means of natural selection. Cambridge, MA: The MIT Press.
[8] David J Montana. Strongly typed genetic programming. In Evolutionary Computation 3-2, 1995, pp 199-230.
[9] David E. Goldberg. Genetic algorithms in search, optimization and machine learning. Addison-Wesley Pub. Co. 1989. ISBN: 0201157675.
[10] Aymeric Zils, François Pachet. Extracting automatically the perceived intensity of music titles. Proceedings of 6th International Conference on Digital Audio Effects (DAFX03), London, UK, September 8-11, 2003.
[11] A.L. Berenzweig, Dan P. W. Ellis. Locating singing voice segments within music signals. IEEE workshop on applications of signal processing to acoustics and audio (WASPAA01), Mohonk NY, October 2001.
[12] Fukunaga, K., "Statistical pattern recognition", Academic press, 1990.