ANALYTICAL FEATURES FOR THE CLASSIFICATION OF PERCUSSIVE SOUNDS: THE CASE OF THE PANDEIRO

Pierre Roy, François Pachet, Sergio Krakowski

Sony CSL Paris
Paris, France
roy@csl.sony.fr
Pachet@csl.sony.fr
skrako@gmail.com

ABSTRACT

There is an increasing need for automatically classifying sounds for MIR and interactive music applications. In the context of supervised classification, we describe an approach that improves the performance of the general bag-of-frame scheme without loosing its generality. This method is based on the construction and exploitation of specific audio features, called analytical, as input to classifiers. These features are better, in a sense we define precisely than standard, general features, or even than ad hoc features designed by hand for specific problems. To construct these features, our method explores a very large space of functions, by composing basic operators in syntactically correct ways. These operators are taken from the Mathematical and Audio Processing domains. Our method allows us to build a large number of these features, evaluate and select them automatically for arbitrary audio classification problems.

We present here a specific study concerning the analysis of Pandeiro (Brazilian tambourine) sounds. Two problems are considered: the classification of entire sounds, for MIR applications, and the classification of attacks portions of the sound only, for interactive music applications. We evaluate precisely the gain obtained by analytical features on these two problems, in comparison with standard approaches.

1. ACOUSTIC FEATURES

Most audio classification approaches use either one of these two paradigms: a general scheme, called *bag-of-frames*, or ad hoc approaches.

The bag-of-frame approach ([2], also cited [41]) consists in considering the signal in a blind way, using a systematic and general scheme: the signal is sliced into consecutive, possibly overlapping frames (typically of 50ms), from which a vector of audio features is computed. The features are supposed to represent characteristic information of the signal for the problem at hand. These vectors are then aggregated (hence the "bag") and fed to the rest of the chain. First, a subset of available features is identified, using some feature selection algorithm. Then the feature set is used to train a classifier, from a database of labeled signals (training set). The classifier thus obtained is then usually tested against another database (test set) to assess its performance.

The use of the features as input to classifiers plays two roles: a dimension reduction role, and a representation role. Indeed, the signal itself could in principle be used as input to classifiers, but its dimension (number of samples) is usually too high with respect to the training set size, resulting in overfitting. Additionally, the time/amplitude representation of signals has long been acknowledged to be poorly adapted to represent perceptive information: audio features used in the classification literature aim precisely at capturing essential perceptive characteristics of audio signals that are not easily revealed in the temporal representation. A source of audio features is for instance MPEG7-audio ([15] or more specifically [28] or [20]) for the music domain. These features are usually of low dimensionality, and contain statistical information from the temporal domain (e.g. Zero-crossing rate), spectral domain (e.g. SpectralCentroid), or more perceptive aspects (such as sharpness, relative loudness, etc.).

The bag-of-frame approach has been used extensively in the MIR domain, for instance by [32]. A large proportion of MIR related papers has been devoted to studying the details of this chain of process: feature identification [28]; feature aggregation [34]; feature selection [26],[22],[7]; classifier comparison or tuning [1],[41].

An even larger proportion of ISMIR papers discuss the application of this approach to specific musical problems: genre classification [38],[21],[25],[39]; orchestral sound [27]; percussion instrument [37],[35],[13],[36]; tabla strokes [9],[6]; audio fingerprinting [5]; noises [12] as well as identification tasks, such as vocal identification [18] or mood detection [19].

This approach achieves a reasonable degree of success on some problems. For instance, speech music discrimination systems based on the bag-of-frame paradigm yield almost perfect results. However, the approach shows limitations when applied to more "difficult" problems. Although classification difficulty is hard to define precisely, it can be noted that problems involving classes with a smaller degree of abstraction are usually much more

difficult to solve. For instance, genre classification works well on abstract, large categories (Jazz vs. Rock), but performance degrades for more precise classes (e.g. Be-bop vs. Hard-bop).

In these cases, the natural tendency is usually to look for *ad hoc* approaches, which aim at extracting "manually" from the signal the characteristics most appropriate for the problem at hand, and exploit them accordingly. This can be done either by defining ad hoc features, integrated in the bag-of-frame approach (e.g. the 4-Hertz modulation energy used in some speech/music classifiers, [32], or by defining completely different schemes for classifying, e.g. the analysis-by-synthesis approach designed for drum sound classification [45], and further developed by [44] and [31].

One of the possible reasons for the limitation of bag-of-frame approach is that the generic features used, such as the Mpeg-7 feature set, do not always capture the relevant perceptive characteristics of the signals to be classified. Some classifier algorithms, such as kernel methods [33] including Support Vector Machines [4],[34] do try to transform the feature space with the aim of improving inter-class separability. However, the increasing sophistication of feature selection or classifier algorithms cannot compensate for any lack of information in the initial features set.

Although ad hoc approaches may indeed reach interesting performance, they are rarely reusable: ad hoc features are, by definition, problem specific. Consequently the scientific contribution (and epistemological status) of reports of ad hoc approaches is highly debatable.

In this work we try to extend the range of applications for which the general bag-of-frame approach gives satisfactory results, by proposing a mechanism that invents specific ad hoc features, in an automatic way to improve the classification performance.

To find better features than the generic ones, one can find inspiration in the way human experts actually invent ad hoc features. The papers quoted above use a number of tricks and techniques to this aim, combined with intuitions and musical knowledge. For instance, one can use some front-end system to normalize a signal, or pass it through some filter, add pre or post-processing to isolate the (hopefully) most salient characteristics of the signal.

We propose here to automate a process of feature invention, by an algorithm which explores quickly a very large space of ad hoc functions. The functions are built by composing together - in the sense of functional composition - elementary operators. We call these functions analytical because they are described by an explicit composition of functions, as opposed to other forms of signal reduction, such as arbitrary computer programs.

This paper is structured as follows: In Section 2 we introduce the EDS system, designed to create automatically and explore large sets of analytical features. Section 3 is devoted to the description of several experiments to compare the performance of analytical features against generic ones, on two sound classification problems for the Pandeiro (Brazilian percussion instrument): an easy one, for MIR applications, and a more difficult one, for interactive music applications.

2. CREATION OF ANALYTIC FEATURES: THE EDS SYSTEM

EDS – Extractor Discovery System – is developed at the Sony CSL laboratory in Paris [45] to study experimentally the notion of analytical feature for audio signal processing applications.

The EDS system is able to explore efficiently the space of analytical features for arbitrary supervised audio classification problem. A problem is determined by a database of audio samples labelled (usually by hand) with a finite set of classes. The exploration of the space of analytical features is based on various function creation methods from a set of basic operators, considered as elementary. These two aspects are described in the following sections.

2.1. A library of elementary operators

The choice of elementary operators is of course arbitrary. These operators were selected so as to allow the creation of functions with a "reasonable" degree of abstraction, i.e. represent salient perceptive characteristics of the sound with a small number of operators (about 10, see below), while allowing to create new, and possibly relevant functions. These operators are either basic mathematical operations (e.g. absolute value, max, mean) or signal processing operators such as Fourier transforms, filters, *Db*, and spectral operators like *spectralCentroid*, *spectralSkewness*. This library also includes more specifically musical operators such as *Pitch* or *Ltas* (Long Term Average Spectrum). For the sake of reproducibility, we describe in this paper results obtained with the 76 basic operators listed in Annex 1.

If we limit the size of analytical features we create (i.e. the number of operators used in its expression), we explore a finite function space. To give a rough idea of its size: the feature space of features composed of at most 5 operator contains $2{,}5.10^9$ functions. In practice, we explore functions of size at most 10, which represents a space of 5.10^{20} functions. Here are some typical examples of functions generated by EDS:

```
(A) Mean(Mfcc(Differentiation(x),5))
(B) Median(Rms(Split(Normalize(x),32)))
```

The first function (A) computes the average of the 5 first cepstral coefficients of the differentiation of the signal (represented by *x*). The second one (B) computes the mean value (*Median*) of the energy (*Rms*) of successive frames (*split*) of 32 samples long in the normalized signal.

Feature creation is controlled by two mechanisms:

1 – Each basic operator is typed according to the physical dimensions of its arguments. Types avoid creating syntactically meaningless features. For instance, the Fft operator takes as input something of the "time/amplitude" type, and its output type is "frequency/amplitude". EDS can therefore generate *Fft(HpFilter(x))*, but not, e.g., *Fft(max(x))*.

2 – Heuristics allow the system to further avoid creating unpromising functions. E.g. a heuristics penalizes functions with too many repetitions, like *Fft(Fft((Fft(x))))*.

In practice, adding a new basic operator to the library amounts to define 1) corresponding typing rules and 2) heuristics to control the use of this operator (see [24]).

2.2. Creating analytical features

The creation of analytical features by composing elementary operators is based on genetic programming search [16]. The main steps of this search are the following:

1. Construction of an initial population of analytical features, by random compositions of operators.
2. Evaluation: compute each feature on all the training signals, then use a classifier (see Section 2.3) to assess performance.
3. Iteration of the process. The next population is built from the best features found in the current population, to which are added new features obtained using various genetic transforms of the current features.

This genetic procedure explores parts of the infinite set of all analytical functions composed of basic operators. The convergence towards "meaningful" or "interesting" analytical features is not guaranteed as this heuristic-based approach can be entrapped into local minima.

The genetic transforms of step 3 are the following:

- *Substitution:* replacing one operators by another one with a compatible type. E.g.

```
(A') Max(Mfcc(Differentiation(x),5))
```

is a substitution (`Max` replaces `Mean`) of (A)

- *Cloning:* special case of substitution which consists in copying a feature but changing its parameters, e.g. :

```
(B') Median(Rms(Split(Normalize(x),64)))
```

is a clone of (B).

- *Mutation*: an extension of substitution to sub expressions appearing in the definition of a feature, which satisfies the typing rules:

```
(A") Mean(Chroma(Normalize(x)))
```

is a mutation of (A): sub expression `Chroma (Normalize(x))` replaces `Mfcc (Differentiation (x),5)`.

- *Crossover*: combining two features to create a new one while satisfying the typing rules. For instance:

```
(C) Mean(Rms(Split(Normalize(x),32)))
(C') Median(Rms(Split(Differentiation(x))))
```

are crossovers between (A) and (B).

- *Addition*: adding an operator to the root of a feature:

```
                    (B")
Abs(Median(Rms(Split(Normalize(x),32))))
```

is an addition of (B).

2.3. Evaluation of features

To evaluate features, we need a computable criterion which measures the quality of a feature, i.e. its capacity to distinguish elements of different classes (labels). There are various ways to define such a criterion. The Fischer Discriminant Ratio [8] is often used because it is simple to compute and reliable for binary problems (two classes). However it is notoriously not adapted to multi-class problems, in particular for non convex distributions of data.

To improve feature evaluation, we chose to implement a "wrapper approach" to feature selection: features are evaluated using a classifier built during the feature search. The fitness is the performance of a classifier built with this unique feature (or more precisely its F-measure [30]) trained on the training database. This measure yields better performance than the Fischer criteria on multi-class problems.

3. PANDEIRO SOUND CLASSIFICATION

The Pandeiro is a Brazilian frame drum (a type of tambourine) used in particular in Brazilian popular music (samba, côco, capoeira, chôro). As it is the case for many popular music instruments, there is no official method for playing the Pandeiro. However, the third author, a professional Pandeiro player, has developed such a method, as well as a notation of the Pandeiro, that we use in this paper. This method is based on a classification of Pandeiro sounds in exactly six categories (see Figure 1):

Tung: Bass sound, also known as open sound;

Ting: Higher pitched bass sound, also open;

PA (big pa): A slap sound, close to the Conga slap;

pa (small pa): A medium sound produced by hitting the Pandeiro head in the center. Also considered as a slap, but softer;

Tchi: The jingle sound;

Tr: A tremolo of jingle sounds.

The need for automatically analyzing Pandeiro sounds is twofold. First, MIR applications, for education notably, require the ability to automatically transcribe Pandeiro solos.

Figure **1**. *The gestures to produce the six basic Pandeiro sounds.*

The second need is more original, and consists in developing real time interaction systems that expand the possibilities of the percussionist, to allow him to increase its musical "powers". In this case, we need to analyze robustly and quickly Pandeiro sounds, to trigger various events (see, e.g.[17]).

We therefore define two different analysis problems, corresponding to these two applications.

The first problem consists in classifying complete sounds (150ms duration) in the 6 classes. The second problem, much more difficult but more useful for real time applications, consists in classifying sounds using the least possible information, typically only the attack (about 3ms, that is 128 samples at 44 kHz), so as to allow a subsequent triggering of a musical event. To this aim we must build a reliable and very fast classifier.

3.1. Available sound databases

We have recorded a 2448 complete Pandeiro sounds (408 of each 6 types). They were produced with the same instrument and recorded on a Shure Beta 98 microphone linked to a MOTU Traveller sound card.

In order to classify the sounds, it is important to finely locate them in time. To this aim, we designed a robust attack identifier, which works as follows, on the sounds of the two databases.

We first extract an auditory spectrogram for the incoming signal [14]. Because of real-time constraints, we only compute an approximation of this spectrogram, as follows. The incoming signal is divided in non-overlapping frames of 1.4ms (64 samples at 44kHz). A loudness value is computed for each frame, generating the "loudness curve". We compute the differentiation of this curve. We call these two curves, the *loudness*" and the *differential*. Both are low pass filtered to reduce noise.

The attack detection is then performed in two phases. First we determine a threshold value for distinguishing actual sounds from noise. To this aim, the player captures 5 seconds of ambient noise (typically room noise as well as soft Pandeiro tchi sounds) and calculate the above mentioned curves from this audio information. The maximum value of these curves define the loudness and differential thresholds.

In the second phase, an attack is reported if, at a certain frame, the loudness level is greater than the loudness threshold and the norm of the differential curve exceeds the differential threshold. This frame is considered as the "attack frame".

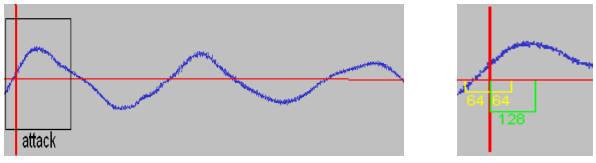

Figure **2**. *The attack detector: on the left, the full sound and attack portion. On the right, a zoom of the pre-attack and post-attack portions of the signal.*

When an attack is reported, two audio files are recorded. The first file is the audio contained both in the attack frame and its preceding frame. This file populates the pre-attack database (see **Figure 2**). We record another audio file with the audio stream right after the attack (the attack frame and one after it). This file populates the post-attack database.

Classifying the sound using only the pre-attack database information is the most difficult and useful problem in our context. The results on the post-attack database are slightly better, as it will be discussed, but they require an extra delay of 1.4ms (to get the next 64 samples) before processing.

3.2. Experiments: training and testing bases

In order to assess the efficiency of analytical features, we compare them to results obtained with a "reference feature set", whose complete list is given in Annex 2. This reference set includes general features commonly used in audio signal classification tasks, and well defined mathematically. The list includes notably the Mpeg-7 audio list, as well as several others, such as *Chroma*, often used for music analysis [10].
We systematically evaluate the performance of two classifiers: one built with the reference set, the other built with the features found by EDS with the set of basic operators in Annex 1.

Each experiment is in turn divided in two parts. First, classifiers are trained on training samples and tested on the test samples. To this aim, databases are systematically divided in two parts, 2/3 for the training, and 1/3 for the test. The samples are chosen randomly, to avoid artifacts (e.g. evolution of the membrane during the recording session, small variations in the player gestures).

In the second part, classifiers are trained and tested only on the test database, using 10-fold cross-validation.

This double experiment aims at showing that the advantages obtained by analytical features are consistent, and do not depend on the conditions of experiments. The cross-validation using only the test database is motivated by the fact the EDS already uses the training database for evaluating the analytical features. So reusing it for training the classifiers could produce biases (although we are not sure why and how).

Finally, for the attack problem, we build an experiment in which the signal itself is used as a feature (this is possible because these signals are very short). The aim is to confirm that the signal is not a good feature.

3.3. Choosing the classifiers

There is a vast literature on supervised learning algorithms [41] West, K., Cox, S., *Features and Classifiers for the automatic classification of musical audio signals*, ISMIR 2004.

[42] West, K., Cox, S., *Features and Classifiers for the automatic classification of musical audio signals*, ISMIR 2004.

[43] with no clear winner in general. To demonstrate the advantages of analytical features, we have conducted experiments with various classifiers, to avoid biases (e.g. SVM, *k*NN, J48). For the sake of clarity, we report here only the results with Support Vector Machines [34], which turned out to be the best and most stable algorithms tried. (We use the implementation provided in Weka [40] with the polynomial kernel.)

We used EDS in a fully automated way for the creation and selection of analytical features. For each problem, we ran the genetic search until no improvements were found in feature fit-

ness. For the complete sound problem, EDS evaluated about 40,000 features. For the attack problem EDS evaluated about 200,000 features.

3.4. Feature Selection

To compare the two approaches (general versus analytical features) in a fair manner, it is important to train classifiers on spaces with identical dimension. For the full sounds, all reference features (cf. Annex 2) could be computed, yielding a feature set of dimension 100. We have therefore selected 100 scalar analytical features among the 23,200 computed by EDS.

In the case of attacks, not all reference features were computable, because there is insufficient data: only 17 reference features could be computed and evaluated, with a total dimension of the feature set of 90. We therefore selected 90 analytical features among the 77,500 (resp. 53,500) EDS created for pre-attacks (resp. post-attacks).

To illustrate the results obtained, we have tried two different feature selection methods. Feature selection is important to avoid using redundant features. Here again, there are many feature selection methods [11] and the choice of the method turns out to be important for the final evaluation of the classifier. To avoid bias, we use, here also, two methods. The first is the IGR algorithm (Information Gain Ratio) [29]. Technically, this corresponds to the Weka *AttributeSelection* algorithm with the following parameters: the *evaluator* is a *InfoGainAttributeEval* and the *search* is a *Ranker*, which allows us to determine a priori the dimension of the feature set.

Secondly, we also developed a feature selection algorithm more suited to the application of EDS to multi-class problems. The idea is to select a feature set that "covers" optimally the classes to learn, from the viewpoint of individual features, that is, essentially of their F-measure (see Section 2.3). This algorithm iterates over all classes and selects successively features with the best F-measure for a given class.

Finally, we present results obtained for various sizes of feature sets (from 1 to 100). This is an important aspect in the context of real-time systems, where we want to minimize the number of features to compute in real time. As we will see, EDS finds not only better features but also feature sets of lesser dimension.

3.5. Results and comments

The tables Figure 3, Figure 4 and Figure 5 show the results obtained:

For the two problems, analytical features found by EDS improve the classification performance. The full sound problem is relatively easy. The use of the full reference feature set (dimension 100) yields a precision of about 99,9%. With the same dimension, analytical features yields the same precision. The gain becomes interesting if we consider feature sets of smaller dimension: 2 analytical features yield a precision of 89,5% versus 78% for general features.

Experiment Description			Feature Set Dimension										
			100	90	75	50	25	15	10	5	3	2	1
Reference	*IGR*	*Train/Test*	99,9	99,9	99,6	99,5	99	99,5	99,1	92,8	88,5	65,2	56
Reference	*IGR*	*10-fold XV*	99,9	99,5	99,5	99,5	99,1	98,6	98,4	92	82	60,5	59,3
EDS	*IGR*	*Train/Test*	99,9	99,9	98,5	98,3	98,9	98,3	99,1	98	68,9	36,1	36,9
EDS	*IGR*	*10-fold XV*	99,9	99,9	99,9	98,8	98	98,4	98,2	97,8	64,7	36	21,2
Reference	*EDS FS*	*Train/Test*	99,9	99,9	99,9	99,8	99,1	99,1	98,9	98,8	93,6	80,8	67,2
Reference	*EDS FS*	*10-fold XV*	99,9	99,6	99,6	99,4	98,6	98,4	98,8	98,3	93,4	78,1	61,6
EDS	*EDS FS*	*Train/Test*	99,9	99,9	98,9	99,9	99,9	99,6	99,5	99	89,9	88,8	73,8
EDS	*EDS FS*	*10-fold XV*	99,9	99,9	98,9	99,7	99,6	99,5	99,4	99	91,3	89,5	73,6

Figure **3**. *Results on full sounds.* ***IGR*** *stands for Information Gain Ratio.* ***EDS FS*** *denotes our feature selection algorithm based on the F-measure.* ***Train/Test*** *denotes the experiment in which the classifier is trained on the training database and tested on the test database.* ***10-fold XV*** *denotes the 10-fold cross validation experiment on the test database.*

Experiment Description			Feature Set Dimension									
			90	75	50	25	15	10	5	3	2	1
Reference	*IGR*	*Train/Test*	94,8	95,6	93	76,8	76	76,1	73,5	65,9	54,2	44,6
Reference	*IGR*	*10-fold XV*	94,8	94,8	92,7	78,8	73,2	72,2	66,2	65,2	48,3	43,3
EDS	*IGR*	*Train/Test*	94,8	95,6	92,9	81	76,6	76	69,6	65,7	54,2	44,5
EDS	*IGR*	*10-fold XV*	95,1	94,5	92,8	78,8	73,2	73,5	66,8	65,3	50,9	45
Reference	*EDS FS*	*Train/Test*	94,7	94,7	94,8	92,4	90,8	88,7	87,2	84,1	71,4	52,4
Reference	*EDS FS*	*10-fold XV*	95,4	94,7	94	91,9	90,8	87,9	85,7	81,5	68,5	51,3
EDS	*EDS FS*	*Train/Test*	96	95,5	95,1	93,9	93,5	93,4	93	89,9	86,2	71,7
EDS	*EDS FS*	*10-fold XV*	95,1	95	95,2	93,3	92,9	92,5	92,5	88,3	84,8	71,6
Signal			75.8	75.8	72.5	67.6	67.1	46	44	36.6	37	35.5

Figure **4**. *Results obtained with on pre-attacks. See above for abbreviations. The "Signal" line gives the performance of classifiers using the input signal directly as a feature.*

Experiment Description			Feature Set Dimension									
			90	75	50	25	15	10	5	3	2	1
Reference	*IGR*	*Train/Test*	91,8	91,3	89,6	76,6	78,3	67,5	64,3	56,1	51,1	49
Reference	*IGR*	*10-fold XV*	92,6	91,2	88,8	79,9	73,2	67,4	64,7	44,2	42,4	34,5
EDS	*IGR*	*Train/Test*	95,1	93,3	92,3	77,7	72,5	63	61,3	54,7	54,5	56,9
EDS	*IGR*	*10-fold XV*	94,9	93,8	92,4	80,8	78,9	62,4	61	55,1	55,9	54,9
Reference	*EDS FS*	*Train/Test*	91,9	91,5	91	87,7	86,7	83,4	83,6	71,7	55,6	43,9
Reference	*EDS FS*	*10-fold XV*	91,9	91,5	90,2	86,1	85,2	78,9	82	68,5	48,6	39
EDS	*EDS FS*	*Train/Test*	94,9	94,4	94	92,1	91,4	87,9	90,1	88,6	80,4	72,1
EDS	*EDS FS*	*10-fold XV*	94,5	94	93,3	91,4	91,4	89	89,5	88	80,1	69,2
Signal			77.7	76.9	73.3	64.1	64.2	60	59.2	58.1	57.5	44

Figure **5**. *Results obtained with on post-attacks. See above for abbreviations.*

The attack problems are more difficult and interesting. Analytical features are still better than general ones, in particular for small feature sets. For the post-attack problem, 3 analytical features perform as well as the 50 best general features.

We can note that the gain evolution depends on the feature selection algorithm used. The standard IGR algorithm does not select the best EDS features for small size feature sets (this result is already known, see [3]). However, our feature selection algorithm yields better results for all sizes of the feature set, as illustrated in Figure 6. This result shows again, if needed, the difficulty in interpreting the precision of classifiers directly.

The performance gain brought by analytical features for small feature sets has a lot of advantages, in particular for real-time applications. For the attack problem, 3 features yield a precision greater than that obtained with 50 reference features. These features are the following:

Abs (*Log* (*Percentile* (*Square* (*BpFilter* (x, 764, 3087)), 64)))

Centroid (*MelBands* (*Differentiation* (*HpFilter* (*Power* (*Normalize* (x), 3), 100)), 6))

Abs (*Sum* (*Arcsin* (*Mfcc* (*Hann* (*HpFilter* (x, 19845)), 20))))

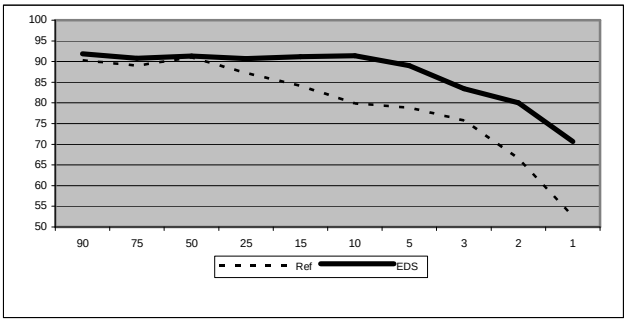

Figure **6**. *Analytical vs. reference features on attacks*

This particular result allows us to consider real-time implementations: on a 3GHz Pentium IV PC, the computation of the 3 features for a 2,8 ms signal takes about 3 ms, to be compared to the computation of 50 generic features, which takes 12 ms, that is 4 times slower.

4. CONCLUSION

We have presented a method for creating audio features, called analytical, by composing basic signal operators, to improve the performance of classification algorithms. We have illustrated this idea on audio classification problems dealing with Pandeiro sounds. In all cases (classifying full sounds, or only portions of the attacks) analytical features do improve the performance of classification, as compared to results obtained with generic, Mpeg-7 like features, in a bag-of-frame approach. The gain is notable both in terms of classification precision and feature set size. Moreover, analytical features improve classifi-

cation algorithms independently of any other optimization process (such as boosting, bagging or *ad hoc* approaches).

5. ACKNOWLEDGMENTS

The work of Sergio Krakowski is partially supported by a CAPES scholarship.

6. REFERENCES

[1] Aucouturier, J.-J. and Pachet, F. Tools and Arc*hitecture for the Evaluation of Similarity Measures : Case Study of Timbre Similarity*. ISMIR 2004.

[2] Aucouturier, J.-J, Defreville, B. and Pachet, F. The bag-of-frame approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music. Journal of the Acoustical Society of America, 2007.

[3] Blum , A. and Langley, P. *Selection of Relevant Features and Examples in Machine Learning*, Artificial Intelligence, 1997 pp.245-271, Dec. 1997.

[4] Boser, B. Guyon, I. and Vapnik V. *A training algorithm for optimal margin classifiers*. In D. Haussler, editor, 5th Annual ACM Workshop on COLT, pp.144-152, Pittsburgh, PA. ACM Press. 1992.

[5] Cano, P., Batlle, E., Kalker, T., and Haitsma, J. *A Review of Audio Fingerprinting*. J. VLSI Signal Process. Syst. 41, 3 (Nov. 2005), 271-284. 2005.

[6] Chordia, P. Segmentation and Recognition of Tabla Strokes, ISMIR, pp. 107-114, 2005.

[7] Fiebrink, R. and Fujinaga, I. *Feature Selection Pitfalls and Music Classification*. ISMIR 2006, pp. 340-341, 2006.

[8] Fisher, R. A. *The use of Multiple Measurements in Taxonometric Problems* Ann. Eugenics, vol. 7, pp. 179-186. 1936.

[9] Gillet, O. and Richard, G. *Automatic Labelling of Tabla Signals*. ISMIR 2003.

[10] Gomez Gutierrez E. *Tonal Description of Music Audio Signals*, PhD Thesis, Universitat Pompeu Fabra, Barcelone, 2006.

[11] Guyon, A. Elisseeff, *An Introduction to Variable and Feature Selection*, Journal of Machine-Learning Research, 3 1157-1182, 2003.

[12] Hanna, P., Louis, N., Dessainte-Catherine, M. and Benois-Pineau, J. *Audio Features for Noisy Sound Segmentation*. ISMIR 2004, 2004.

[13] Herrera, P., Yeterian, A. and Gouyon, F. *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.

[14] Jehan, T. *Hierarchical Multi-Class Self Similarities* IEEE WASPAA, 2005.

[15] Kim, H.G., Moreau, N. and T. Sikora Mpeg7 *Audio and Beyond: Audio Content Indexing and Retrieval*. Wiley & Sons. 2005.

[16] Koza, J. R. Genetic Programming: on the programming of computers by means of natural selection, Cambridge, MA: The MIT Press. 1992.

[17] Krarkowski, S. Pandeiro+, music video available at: http://www.skrako.com/eng/pop_video.html?aguas

[18] Lay Nwe, T. and Wang, Y. *Automatic Detection Of Vocal SegmentsiIn Popular Songs*. ISMIR 2004.

[19] Liu, D., Lu, L. and Zhang, H.J., *Automatic mood detection from acoustic music data*, ISMIR 2003.

[20] McEnnis, D. McKay, C., Fujinaga, I. Depalle, P. *jAudio: a feature extraction library*, Ismir 2005.

[21] McKay, C. and Fujinaga, I. Automatic Genre Classification Using Large High-Level Musical Feature Sets. ISMIR 2004.

[22] McKinney, M.F. and Breebart, J. *Features for audio and music classification.* ISMIR 2003.

[23] Meng, A. and Shawe-Taylor, J. *An Investigation of Feature Models for Music Genre Classification Using the Support Vector Classifier*. ISMIR 2005.

[24] Pachet, F. and Zils, A. *Automatic Extraction of Music Descriptors from Acoustic Signals*. ISMIR 2004.

[25] Elias Pampalk, Arthur Flexer & Gerhard Widmer Improvements of Audio-Based Music Similarity and Genre Classification (pp. 628-633), ISMIR 2005

[26] Peeters, G. and Rodet, X. *Automatically selecting signal descriptors for sound classification*. Proceedings of the 2002 ICMC, Goteborg (Sweden). 2002.

[27] Peeters, G. Automatic Classification of Large Musical Instrument Databases Using Hierachical Classifiers with Inertia Ratio Maximization, 115th AES Convention New-York, NY, USA, 2003.

[28] Peeters, G. *A large set of audio features for sound description in the Cuidado project*. http://recherche.ircam.fr/equipes/analyse-synthese/peeters/ARTICLES/Peeters_2003_cuidadoaudiofeatures.pdf

[29] Quinlan, J.R. *C4.5: Programs for machine learning*. Morgan Kaufmann. 1993.

[30] Rijsbergen, C. J. van *Information Retrieval*. Butterworths, London. 1979

[31] Sandvold, V. Gouyon, F. Herrera, P. Percussion classification in polyphonic audio recordings using localized sound models, ISMIR 2004.

[32] Scheirer, Eric D. and Slaney, Malcolm *Construction and evaluation of a robust multifeature speech/music discriminator*. Proc. ICASSP '97. 1997.

[33] Schölkopf, B. and Smola, A. *Learning with Kernels*, MIT Press, Cambridge, MA. 2002.

[34] Shawe-Taylor, J. and Cristianini, N. Support Vector Machines and other kernel-based learning methods - Cambridge University Press. 2000.

[35] Sinyor, E., McKay, C., Fiebring, R., McEnnis, D. and Fujinaga, I. *Beatbox Classification Using ACE*. ISMIR 2005: 672-675, 2005.

[36] Steelant, D. van Tanghe, K. Degroeve, S. De Baets, B. Leman, M. and Martens, J.-P. *Classification of percussive sounds using Support Vector Machines*, Proceedings of the annual machine learning conference of Belgium and The Netherlands, 2004.

[37] Tindale, A.R., Kapur, A., Tzanetakis, G. and Fujinaga, I. *Retrieval of percussion gestures using timbre classification techniques*, ISMIR 2004,2004.

[38] Tzanetakis, G. *Automatic Musical Genre Classification of Audio Signals*, ISMIR 2001.

[39] Tzanetakis, G. Cook, P. *Musical Genre Classification*, IEEE Transactions on Speech and Audio Processing, Vol. 10, No. 5, July, pp. 293-301. 2002.

[40] Weka, Data Mining Software in Java, see http://www.cs.waikato.ac.nz/ml/weka/

[41] West, K., Cox, S., *Features and Classifiers for the automatic classification of musical audio signals*, ISMIR 2004.

[42] West, K., Cox, S., *Features and Classifiers for the automatic classification of musical audio signals*, ISMIR 2004.

[43] Witten, I.H. Eibe, F. *Data Mining: Practical Machine Learning Tools and Techniques.* M. Kaufmann Publisher, 2nd Edition. 2005.

[44] Yoshii, K., Goto, M., and Okuno, H.G. AdaMast: *A Drum Sound Recognizer based on Adaptation and Matching of Spectrogram Templates*, ISMIR 2004.

[45] Zils A., Pachet F., Delerue O., Gouyon F. *Automatic Extraction of Drum Tracks from Polyphonic Music Signals*. Proceedings of WEDELMUSIC, Dec. 2002.

[46] Zils, A. *Extraction de descripteurs musicaux: une approche évolutionniste*, PhD, Univ. Paris 6. 2004.

7. ANNEXES

All the sounds and results of this study are made available to interested readers, as well as feature files (Weka format): http://SecondAuthorWebSite/pandeiro

7.1. Annex 1 – Basic EDS operators

The list of basic operators used by EDS in this study is the following:

Abs	HarmSpectralVar	PeakPos
Arcsin	HFC	Percentile
AttackTime	HMean	Pitch
Autocorrelation	HMedian	PitchBands
Bandwidth	HMax	Power
BarkBands	HMin	Range
Bartlett	HpFilter	RHF
Blackman	Integration	Rms
BpFilter	Inverse	SpectralCentroid
Centroid	Iqr	SpectralDecrease
Chroma	Length	SpectralFlatness
Correlation	Log10	SpectralKurtosis
dB	LpFilter	SpectralRolloff
Differentiation	Max	SpectralSkewness
Division	MaxPos	SpectralSpread
Envelope	Mean	Split
Fft	Median	SplitOverlap
FilterBank	MelBands	Sqrt
Flatness	Min	Square
Hamming	Mfcc0	Sum
Hann	Mfcc	Triangle
Hanning	Multiplication	Variance
HarmSpectralCentroid	Normalize	Zcr
HarmSpectralDev	Nth	Harmonicity(Praat)
HarmSpectralSpread	NthColumns	Ltas(Praat)

A precise description of operators can be found in [46].

7.2. Annex 2 – Reference features

The list of general features used as the reference set is the following (features preceded by '*' could not be computed on the attack sounds because of their size):

```
* HarmonicSpectralCentroid(Hanning(x))
* HarmonicSpectralDeviation(Hanning(x))
* HarmonicSpectralSpread(Hanning(x))
Log10(AttackTime(x))
*Pitch(Hanning(x))
SpectralCentroid(Hanning(x))
* SpectralFlatness(Hanning(x))
SpectralSpread(Hanning(x))
Centroid(x)
PitchBands(Hanning(x),12.0)
Mfcc0(Hanning(x),20.0)
* HarmonicSpectralVariation(SplitOverlap(Hanning(x),2048,0.5))
Rms(x)
RHF(Hanning(x))
HFC(Hanning(x))
SpectralKurtosis(Hanning(x))
SpectralSkewness(Hanning(x))
SpectralRolloff(Hanning(x))
Iqr(x)
Chroma(Hanning(x))
MelBands(Hanning(x),10.0)
BarkBands(Hanning(x),24.0)
Zcr(x)
```