Classification of Percussive Sounds Using Support Vector Machines

D. Van Steelant[1], K. Tanghe[2], S. Degroeve[3], B. De Baets[1], M. Leman[2], J.-P. Martens[4]

[1]Department of Applied Mathematics, Biometrics and Process Control, Ghent University, Belgium
[2]Department of Musicology (IPEM), Ghent University, Belgium
[3]Department of Bioinformatics, Ghent University, Belgium
[4]Department of Electronics and Information Systems (ELIS), Ghent University, Belgium

dirk.vansteelant@UGent.be

1 Introduction

With the explosive growth of the amount of digital music available on the Internet, Musical Information Retrieval (MIR) has become a topic that has attracted the attention of researchers in a wide range of disciplines. The quality of a content-based retrieval system depends heavily on how well the individual components for representation and matching of the data perform. Most existing commercial music databases use text as the main supplier of meta-data of music, such as the name of the artist/performer and the title of the song. For text, rapid matching methods are available and are applied extensively in search engines on the World Wide Web. However, once such meta-data is incomplete or not available, all of the existing commercial systems will fail to deliver.

1.1 The MAMI-project

In order to integrate the necessary expertise to build a MIR system, the MAMI-project (Musical Audio MIning) (Leman, 2002) aims at working out methodologies, techniques and software tools for content-based musical audio mining by bundling the efforts of musicologists, engineers, mathematicians and computers scientists. MAMI is centered on the 'query-by-imitation' paradigm, where users can retrieve a musical piece by means of its sound characteristics, either by describing, playing or vocally imitating the piece.

1.2 Motivation

In order to supply a ranked list of candidate songs to the user, the system has to match an intermediate representation of the (melodic or rhythmic) input with a similar representation for all the songs in the database; this will typically be done by means of a (time-consuming) dynamic programming technique. To speed up the query any additional information that can narrow the search space is welcome; not only meta-data but also a description of the content of the target song or the musical genre to which it belongs can be used for this purpose.

A user study (Lesaffre, 2003) has shown that when users are asked to imitate a song they are familiar with, some of them will reproduce the rhythmic structure of the piece. This is one of the motivations for analyzing the percussive content of musical audio; if a transcription can be obtained, it can be matched with the description delivered by the user, used as a feature for genre classification or provide valuable information for the determination of beat, tempo and rhythmic structure. Although important classes of music do not contain any percussion, if a transcription system is able to establish this absence, this information can still be used for the determination of style and/or genre.

For the recognition of drum sounds three levels of difficulty can be defined:

- Isolated drum sounds
- Overlapping drum sounds
- Overlapping drum sounds layered with other instruments

Classification of percussive sounds belonging to the first level can for example be employed in the organization of sample libraries. Techniques able to classify sounds belonging to the second level can be used if a multi-track recording of the piece is available or for the analysis and/or manipulation of drum loops and break beats. The greater part of existing music is only available as a stereo track recording and needs to be analyzed by means of techniques addressing the third level. Although in this paper we will mainly deal with problems situated at the second level, we will give some promising results on the analysis of percussion mixed with (synthetic) music. Another motivation for concentrating on the second level is that recently (Virtanen, 2001) it was shown possible to extract drum tracks from musical audio by subtracting the harmonic parts from the signal.

Obtaining a full transcription of the percussive content of musical audio is a challenging task and, to our best knowledge, has never been attempted using SVM. We will therefore concentrate on two important classes of sounds (omnipresent in Western popular music): bass drums (typically low-pitched and strongly indicating the beat) and snare drums (with highly noisy components, delivering important clues about the metrical structure of the song).

Support Vector Machines (SVM) (Vapnik, 1995) have not only proven to offer robust classification in various application areas such as text classification and bio-informatics, they also implicitly provide information about the relevance of the features being used. Although feature selection is beyond the scope of this paper, we intend to exploit this extra information in the future. Additionally, if the complexity of the models can be controlled, their ability to classify new instances within a small amount of time makes them well suited for applications in real-time environments such as beat-tracking or automated accompaniment systems.

1.3 Previous Work

A recent overview of classification techniques for musical instrument sounds in general can be found in (Herrera-Boyer, 2003). Percussive instruments represent a special case as they can be considered to be pitch-independent, so their appearance throughout a musical piece is much more constant. Although this makes them good candidates for localization/classification, they only represent a small part of previous research and in most cases only recognition of isolated sounds is investigated.

In (McDonald, 1997) Spectral Centre Trajectories are used to classify percussive sounds but tests are only conducted on a very small database. In (Sillanpää, 2000) short-time energy values in 25 Bark critical bands are used as features for the recognition of drum sounds. Per class four cluster centers (i.e. models) are computed from the training examples using fuzzy c-means clustering and new events are classified by fitting them to the models using a weighted least squares error measure. Only events taking place on an estimated metrical grid are considered and it is not clear how such a setup would cope with music that contains syncopated rhythm or swing. In (Zils, 2002) a percussion transcription is obtained by an analysis by synthesis technique, whereby the sound searched for is gradually synthesized from the signal; a success rate of over 75% is reported. A large-scale study in (Herrera, 2003) uses different subsets of temporal and spectral descriptors (up to 207) for the recognition of thirty different classes of isolated percussion instruments. K-NN, Kernel Density (KD) estimation, Canonical discriminant analysis and decision trees (C4.5) were investigated as classification techniques. KD combined with correlation-based feature selection yielded a 85% hit rate.

1.4 Outline

The rest of this paper is organized as follows: in Section 2 we will describe how training and test data were generated using samples gathered from commercial CD's, standard MIDI songs and sequencer software. Relevant descriptors for audio data will be presented and Support Vector Machines will be formally introduced. In Section 3 we will report results for a series of experiments and in Section 4 we will comment on these results and give directions for future research.

2 Material and Methods

2.1 Selecting Sounds

We have gathered samples belonging to five classes (bass drum, snare drum, hihat, cymbal and tom) from commercial sample CD's. Such CD's typically provide indications on the class to which a sound belongs by the samples name or its location in the directory structure but this information is not always equally reliable. Listening to the sounds we realized that some of them were mixed with other (percussive) instruments and therefore we had them classified by two users; only the samples that were considered to be “pure” and correctly classified by both users were retained. This yielded 656 bass drums, 604 snare drums, 414 hihats, 141 cymbals and 213 toms; in all classes samples of the acoustic as well as electronic type were selected.

2.2 Generating Audio Data

For any supervised learning algorithm annotated data is a necessity. As the manual annotation of musical audio files is cumbersome and time-consuming, we have chosen to generate audio data starting from a transcription which automatically provides appropriate labels for the events taking place in the audio stream.

Some of the audio data for our experiments was created from randomly generated transcriptions expressing which randomly selected sounds are played with random velocity at fixed locations (grid points). The process is guided by a coincidence matrix specifying the probability that two types of percussive sounds are played at the same time; the matrix was set so that bass and snare drum never coincide but can be mixed with any other sound from the remaining classes of samples.

To gather more realistic data, MIDI (Musical Instrument Digital Interface) transcriptions were exploited. MIDI is a widely used communications protocol that allows electronic musical instruments to interact with each other. Standard MIDI assigns classes of instruments to predefined tracks (e.g. drums on track 10) which makes it possible for an electronic sound device supporting standard MIDI to play songs with its own internal sounds. From 32 songs in standard MIDI format we selected 16 measures of the drum track. These 32 files were loaded into a sequencer program. Eight variations for each track were generated by selecting at random pairs of bass and snare drum from the set of samples while the other drum sounds were drawn from a standard MIDI drum set (yielding 256 audio files). All files were saved as mono wave files sampled at 44.1 KHz.

2.3 Onset Detection

A transcription system for music typically consists of a front-end for onset detection, i.e. the localization of time indices where new events occur, followed by a recognizer/classifier. Onset detectors try to locate the starting points of musical events by detecting a more or less sudden increase in the energy of the waveform and/or by detecting abrupt changes in the spectral contents of the sound.

In order not to introduce any errors due to the incorrect localization of events, we did not perform any onset detection but instead used the timing and labelling information available in the transcription files to determine at what position in time features need to be extracted and whether an event is a positive or negative instance for our binary classifiers.

2.4 Descriptors for Audio

Digital audio corresponds to a very high data rate (88 Kbyte/s for mono CD quality); to arrive at a manageable data rate, one needs to select descriptors that capture the characteristics of the audio while suppressing details which are redundant for the problem at hand. This data reduction will typically be done by sliding a window with a fixed step over the raw audio signal (e.g. a 20 ms window every 10 ms) and by computing at every step descriptors over that window.

2.4.1 Fixed Context

The events we are trying to classify do not have a fixed length; the bass drums in our database for example have a duration ranging from 71 ms to 1.892 s. Although SVM are able to handle variable temporal representations by applying specific kernels, e.g. (Shimodaira, 2001), determining the end of an event in musical audio (offset detection) is difficult. We therefore decided to choose a fixed context of 70 ms at the beginning of the events over which descriptors are to be computed. Although this choice is rather arbitrary, experiments show that the information available in these short segments sufficiently captures the characteristics of the events for correct classification.

In order not to confuse the binary classifiers, we excluded any negative examples that lie within the range of 50 ms of a positive example.

2.4.2 Energy-related Descriptors

A first set of features describes the energy in the signal computed by means of a Root Mean Square (RMS) formula. When inspecting the accumulated spectra of hundreds of bass drums, snare drums and hihats, it can be seen that the spectral energy distributions of these different sounds are located in more or less distinct frequency bands (although not completely separated). Hence we divided the spectrum into three frequency bands and computed energy-related features over these bands:

- RMS in the whole signal.
- RMS per frequency band.
- Ratio of RMS per band to overall RMS.
- RMS per band relative to RMS of other bands (1 to 2, 1 to 3 and 2 to 3).

These features were extracted for the whole context, yielding a 10-dimensional vector.

2.4.3 Temporal and Spectral Descriptors

Temporal features are computed on the sample signal; following descriptors were withheld:

- Zero Crossing Rate (ZCR): number of times per second the signal changes sign.
- Crest Factor: ratio of maximum absolute value sample signal to RMS in the segment.
- Temporal Centroid: the center of gravity of the distribution of the absolute values of the samples in the window.

Spectral features are computed from a Fast Fourier Transform (FFT), which converts the time domain data to the frequency domain:

- Spectral Centroid: the center of gravity of the spectrum.
- Spectral Skewness: the third order central moment of the spectrum.
- Spectral Kurtosis: the fourth order central moment of the spectrum.
- Spectral Rolloff: The value R such that

  $$\sum_{i=1}^{R} M[f_i] = 0.85 \sum_{i=1}^{N} M[f_i]$$

  where $M[f_i]$ is the amplitude for the frequency at bin i and N is the number of frequency bins.

All features were extracted for the whole context length, adding another 7 dimensions to the input space.

2.4.4 MFCC

Logan (2000) shows that Mel Frequency Cepstral Coefficients (MFCC), short-term spectral-based features widely used for speech recognition, are appropriate as a representation for music by examining the functionality of a music/speech discriminator. MFCC are especially interesting for complex music analysis because they combine low-dimensionality and the ability to discriminate between different spectral content. The amount of detail in the description depends on the number of coefficients extracted; for our experiments 12 coefficients were computed.

The temporal deployment of these descriptors is further captured by computing their first and second order derivatives. As a window size of 20 ms and frame step of 10 ms for the extraction of this kind of descriptors is often advised, we used these settings and we computed the mean and standard deviation of the coefficients and their first and second order derivatives over the context, yielding a 72-dimensional vector.

2.5 Training and Test Data

After feature extraction was performed, 2 data sets were constructed, one for Bass Drum (BD) and one for Snare Drum (SD). Next, both data sets were split into a 87.5% training set (for model selection and training) and 12.5% test set; for data extracted from the sequences this was done by random sampling while taking into account the balance between positive and negative instances. For the data extracted from the MIDI files the split was done on file level so that the bass and snare drums in the test set were not used during the training phase.

2.6 Scaling Data

The training data were scaled so that every feature lies within the range of $[-1, 1]$. The scaling parameters for every feature were stored and later on used to scale the test data. This was done in order to imitate the situation in a real-world application where the range of features has to be estimated from the available training data. As a consequence, the features in the test set do not necessarily lie within the range of $[-1, 1]$. Some informal testing showed that this does not compromise the performance of the trained models.

2.7 SVM

Formally, a data set T contains l instances $\mathbf{x_i}$ ($i = 1, \ldots, l$) with each $\mathbf{x_i}$ labelled as y^+ or y^- (known as *classes*), indicating a positive or negative instance, respectively. Each index x_{ij} ($j = 1, \ldots, n$) in vector $\mathbf{x_i}$ is a feature f_j as described above.

The Support Vector Machine is a data-driven method for solving two-class classification tasks. The Linear SVM (LSVM) separates the two classes in T with a hyperplane in the feature space such that:

(a) the "largest" possible fraction of instances of the same class are on the same side of the hyperplane, and

(b) the distance of either class from the hyperplane is maximal.

The prediction of a LSVM for an unseen instance $\mathbf{z}$ is $y^+{=}1$ or $y^-{=}{-1}$, given by the decision function

$$pred(\mathbf{z}) = \mathrm{sgn}(\mathbf{w} * \mathbf{z} + b)\,. \tag{1}$$

The hyperplane is computed by maximizing a vector of Lagrange multipliers α in

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i\, \alpha_j\, y_i\, y_j\, K(\mathbf{x_i}, \mathbf{x_j})\,,$$

subject to:

$$0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^{l} \alpha_i\, y_i = 0\,, \tag{2}$$

where C is a parameter set by the user to regulate the effect of outliers and noise, i.e. it defines the meaning of the word "largest" in (a).

Some tolerance (denoted as ϵ) on the constraints in Equation 2 is acceptable.

The function K is a kernel function and maps the features in T, called the input space, into a feature space defined by K in which then a linear class separation is performed. For the LSVM this mapping is a linear mapping:

$$K(\mathbf{x_i}, \mathbf{x_j}) = \mathbf{x_i} * \mathbf{x_j}\,. \tag{3}$$

The non-linear mapping used in this paper is the Gaussian-SVM (GSVM)

$$K(\mathbf{x_i}, \mathbf{x_j}) = e^{-|\mathbf{x_i} - \mathbf{x_j}|^2/\gamma^2}\,. \tag{4}$$

After calculating the α_i's in (2), the decision function (1) becomes:

$$pred(\mathbf{z}) = \mathrm{sgn}(\sum_{i=1}^{l} \alpha_i\, y_i\, K(\mathbf{x_i}, \mathbf{z}) + b)\,. \tag{5}$$

An instance x_i for which α_i is not zero is called a Support Vector (SV). Note that the prediction calculated in Equation 5 uses the support vectors only. As such, the support vectors are those instances that are closest to the decision boundary in feature space.

2.8 Model Selection

All SVM in our experiments were trained using SVM$^{\text{light}}$ 5.0 (Joachims, 1999)[1] in classification mode with all parameters at their default values, except for C and the kernel-related parameter γ.

We had the optimal parameters for Gaussian kernels (C, γ) established by *looms* (Lee, 2000)[2] using the loose stopping criterion $\epsilon = 0.1$. In the same fashion we estimated an optimal parameter (C) for a linear kernel by training SVM with $\epsilon = 0.1$ and $C = 2^i$ ($i = -8, \ldots, 0, \ldots, 10$) on the training data using the SVM$^{\text{light}}$ option for estimation of the leave-one-out (*loo*) error. For C and γ yielding the lowest *loo*-error estimate a final model was retrained with $\epsilon = 0.001$; in the sequel it will become clear that this strategy failed for specific data sets and we had to resort to N-fold Cross Validation (CV) to select models. Next to classification accuracy, we will report on precision and recall as there is a substantial imbalance between positive and negative instances in the data.

3 Experiments and Results

All data and results can be downloaded at ftp://ipem.UGent.be/MAMI/Public/Data/

3.1 Experiment 1

For our first experiment we generated 1000 random sequences with 5 grid points as described

[1]http://svmlight.joachims.org/

[2]http://www.csie.ntu.edu.tw/˜cjlin/looms/

in Subsection 2.2 using the five classes of drum sounds. The result is a positive/negative ratio of 1564/3433 for BD and 1891/3106 for SD (three of the files ended with a sound too short to compute features over 70 ms, so these examples were discarded).

The 14-dimensional input vectors (energy-related features, ZCR, Spectral Centroid, Kurtosis and Skewness) were computed for all instances and SVM were trained using a linear and Gaussian kernel. The results can be found in Table 1.

	LSVM(a)		GSVM	
	BD	SD	BD	SD
C	4	512	512	4
γ	-	-	0.032	2.048
accuracy	94.07	85.58	95.03	89.26
precision	90.72	82.30	94.09	86.27
recall	90.26	78.81	89.74	85.17
#SV	743	1470	613	1171

Table 1: Experiment 1: classification of the 12.5% test set using LSVM(a) (model selection by *loo*-error estimation) and GSVM.

The use of a Gaussian kernel results in an increase in performance, especially for SD, which seems to be harder to classify (for this data set).

3.2 Experiment 2

For the audio data obtained from MIDI we computed the complete (89-dimensional) set of descriptors. The positive/negative ratio is 12928/26088 for BD and 8552/30720 for SD.

Here, our strategy for LSVM model selection failed; for increasing values of C, the *loo*-error estimate decreased monotonically while computation time increased exponentially. We therefore used N-fold CV to select models; as the training set contains 7 variations for each MIDI file, we set $N = 7$ and created the folds in such a way that BD and SD of the same type are used either for training or testing in each CV loop. In order to combine the obtained average precision and recall we computed

$$F_\beta = \frac{(\beta^2 + 1) \cdot precision \cdot recall}{\beta^2 \cdot precision + recall}$$

with β a user-controlled parameter expressing the preference for either high precision or high recall ($\beta = 1$ in the sequel). The C maximizing the F-measure was used to retrain the final model. Results can be found in Table 2.

	LSVM(b)		GSVM	
	BD	SD	BD	SD
C	0.015625	0.5	2	4
γ	-	-	0.256	0.256
accuracy	93.19	95.70	94.22	96.21
precision	90.94	89.72	88.25	89.04
recall	88.24	90.65	95.24	94.20
#SV	3211	2065	1375	1333

Table 2: Experiment 2: classification of the 12.5% test set using LSVM(b) (model selection by 7-fold CV) and GSVM.

The difference in performance between BD and SD has shifted in favor of SD but is less pronounced.

3.3 Experiment 3

Due to the repetitive nature of drum patterns, it is very likely that much of the information in the data of Experiment 2 is redundant, i.e. many instances represent the same superposition of sounds. We therefore reduced the training set to 10% of its original size by randomly sampling examples while respecting the balance. New scaling parameters were computed and applied to the test set, which kept its original size. As the *loo*-error estimation again failed to converge, we used 7-fold CV for model selection.

	LSVM(b)		GSVM	
	BD	SD	BD	SD
C	0.0625	4	8	32
γ	-	-	0.256	0.064
accuracy	93.36	95.11	95.28	96.19
precision	91.68	90.05	91.40	90.99
recall	87.93	87.18	94.68	91.58
#SV	411	217	556	250

Table 3: Experiment 3 with reduced training set: large decrease of #SV while keeping up performance.

The most remarkable phenomenon (in Table 3) is the substantial reduction of the number of support vectors while only a small decrease in performance is observed (even a small increase in the case of BD). This suggests that while gathering training data some kind of preprocessing can be beneficial; if a transcription is available, identical superpositions of sounds can be discarded and otherwise a cluster analysis can point out which examples are redundant.

3.4 Experiment 4

In order to investigate the applicability of our approach to real musical recordings, we mixed the audio data from Experiment 1 with excerpts of music from MIDI songs (containing no drums). Results in Table 4 show only minor changes in performance, which suggests that our choice of features and classification method is also suitable for the detection of percussion in stereo recordings.

	LSVM(a)		GSVM	
	BD	SD	BD	SD
C	8	8	16	16
γ	-	-	1.024	0.512
accuracy	92.63	85.26	95.19	87.66
precision	88.60	82.14	94.12	84.72
recall	87.69	77.97	90.26	82.20
#SV	789	1699	696	1380

Table 4: Experiment 4 with MIDI audio added.

3.5 Experiment 5

As the data for training and evaluation in Experiment 2 was split at file level, we also analyzed the performance of the classifiers on the individual files. For the GSVM and BD for example, this unveiled some interesting phenomena; for 23 of the 32 files all instances were classified correctly while for the remaining ones either 100% precision or 100% recall was obtained, indicating that on file level errors are either all false positives or all false negatives. Adjusting the bias b in the decision function (5) for individual files in a conservative way (i.e. without introducing new errors) resulted in the correct classification of another 7 files while the errors in the two remaining ones were further reduced, yielding an overall accuracy of 98.56% (96.96% precision, 98.76% recall).

This is a very interesting feature in the case of assisted annotation, a kind of supervised classification where the user can initiate multiple passes through the classification stage and gets (auditive and/or visual) feedback after each pass. This experiment shows that in such a setup, accuracy can be greatly improved by tweaking just one parameter while there is no need for retraining the classifier.

4 Conclusions and Future Work

The results for all of our experiments show that the audio features we have used and SVM classifiers combine well to a technique for the recognition of drum sounds in musical audio. The methodology can be extended for the detection of a wider range of percussive instruments. Although only one experiment treats the situation where other musical instruments are present, results incite further research.

The observation that linear kernels perform only slightly worse than the Gaussian ones is an important finding for applications in time-critical environments. A linear SVM with approximately 1300 support vectors classifies 5000 examples (89-dimensional) in less than 10 ms while it takes a GSVM with the same amount of SV close to 10 s (done on a mobile Pentium III 1.2 GHz with 256 DDR RAM); this difference could turn out to be crucial in a system that, besides classification, also needs to perform onset detection and feature extraction.

The recognition rate can still be improved by gathering more data; this process should be guided by the findings from Experiment 3. MIDI transcriptions of drum tracks contain important clues about which sounds are more likely to coincide and this information can be used for collecting appropriate training sets.

Another interesting topic of research is whether SVM can be employed for onset detection, a problem similar to segmentation as described in (Davy, 2002). Two approaches can be considered; an integrated setup using a single SVM for segmentation and recognition in a single pass or a cascade of two SVM. In the latter case, segmentation should clearly be optimized for high recall as the consecutive classifier can still discard any incorrectly detected events.

As mentioned before, assisted annotation is an interesting application area that needs to be further investigated; once real musical recordings are being used, labelling can be done in a bootstrapping fashion (using a recognizer trained on already available data) and the annotator should have automated methods for correcting errors at his disposal. In such an iterating process, the recognition system should be adaptive in order to learn from previous mistakes without extensive computational efforts for retraining.

Finally, as there is a vast amount of candidate features for the modelling of audio and various ways of encoding them (e.g. the size of the context), future research should try to extend the feature set and at the same time, for the sake of complexity, reduce it by means of feature selection methods (e.g. as in (Degroeve, 2002)). Findings related to what kind of features are relevant for the recognition of percussion would provide interesting feedback to researchers in the field of musicology and perceptual psychology.

Acknowledgements

This research is funded by the Flemish Institute for the Promotion of Scientific and Technical Research in Industry.

References

M. Leman, L. Clarisse, B. De Baets, H. De Meyer, M. Lesaffre, G. Martens, J.-P. Martens and D. Van Steelant, Tendencies, perspectives, and opportunities of musical audio-mining, Journal Revista de Acústica 2002, Vol. 33, No. 3–4.

M. Lesaffre, D. Moelants, M. Leman, B. De Baets, H. De Meyer, G. Martens and J.-P. Martens, User behavior in the spontaneous reproduction of musical pieces by vocal query, In *Proceedings of the 5th Triennial ESCOM Conference* (Hannover, Germany), 2003.

T. Virtanen, Audio signal modeling with sinusoids plus noise, MSc thesis, Tampere University of Technology, 2001.

V.N. Vapnik, The nature of statistical learning theory, Springer-Verlag, 1995.

P. Herrera-Boyer, G. Peeters and S. Dubnov, Automatic classification of musical instrument sounds, Journal of New Music Research 2003, Vol. 32, No. 1, pp. 3-21

S. McDonald and C.P. Tsang, Percussive sound identification using spectral centre trajectories, Technical report, 1997.

J. Sillanpää, A. Klapuri, J. Seppänen and T. Virtanen, Recognition of acoustic noise mixtures by combined bottom-up and top-down processing, In *Proceedings of the European Signal Processing Conference EUSIPCO* (Tampere, Finland), 2000.

A. Zils, F. Pachet, O. Delerue and F. Gouyon, Automatic extraction of drum tracks from polyphonic music signals, In *Proceedings of the* 2^{nd} *International Conference on Web Delivering of Music* (Darmstadt, Germany), 2002.

P. Herrera, A. Dehamel and F. Gouyon, Automatic labeling of unpitched percussion sounds, In *Proceedings of Audio Engineering Society, 114th Convention* (Amsterdam, The Netherlands), 2003.

H. Shimodaira, K. Noma, M. Nakai and S. Sagayama, Support vector machine with dynamic time-alignment kernel for speech recognition, In *Proceedings of Eurospeech* (Aalborg, Denmark), 2001.

B. Logan, Mel frequency cepstral coefficients for music modelling, In *Proceedings of International Symposium on Music Information Retrieval* (Plymouth, MA, USA), 2000, pp. 23–25.

T. Joachims, Making large-scale SVM learning practical, Advances in Kernel Methods - Support Vector Learning (B. Schölkopf, C. Burges and A. Smola, eds.), MIT Press, 1999.

J.-H. Lee and C.-J. Lin, Automatic model selection for support vector machines, Technical report, Dept. of Computer Science and Information Engineering, National Taiwan University, 2000.

M. Davy and S. Godsill, Detection of abrupt spectral changes using support vector machines. An application to audio signal segmentation, In *Proceedings International Conference on Acoustics Speech and Signal Processing* (Orlando, USA), 2002.

S. Degroeve, B. De Baets, Y. Van de Peer and P. Rouzé, Feature subset selection for splice site prediction, Bioinformatics 2002, Vol. 18, No. 2, pp. 75–83.