Unsupervised Generation of Percussion Sequences from a Sound Example

Marco Marchini

Master Thesis MTG - UPF / 2010
Master in Sound and Music Computing

Master Thesis Supervisor:
Hendrik Purwins
Department of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona

UNIVERSITAT POMPEU FABRA

"Music is the pleasure
the human soul experiences from counting…
…without being aware that… it is counting!"

Gottfried Wilhelm Leibniz
(1646 - 1716)

Abstract

In this masters project, a system is developed for the analysis of the structure and the style of a percussive audio sequence with the aim of generating an arbitrarily long musically meaningful and interesting sound sequence with the same stylistic characteristics as the original. The framework is developed around a Variable Length Markov Chain (VLMC) that is used to predict and re-shuffle the basic bricks of a musical phrase, the musical events. Using an onset detector, at first, the sound is split into segments. Applying various clustering thresholds simultaneously, the clustering of the segments yields a multi-level discrete representation of the sound sequence. Then a regularity estimation of the levels is performed to determine the levels of refinement where regular time patterns appear. Those periodic events are then used as the core structure on top of which a regular temporal grid is built. The temporal grid defines the events, the tempo, and the meter. Finally different resolution levels are taken into account for the prediction using a generation strategy enhancing statistical significance while maximizing cluster resolution in order to avoid musical discontinuities. On a data base of percussion and beat boxing examples, the system was evaluated objectively as well as subjectively by two professional percussionists, showing that the automatically generated sequences maintain the style and beat of the original and that they are musically interesting.

Introduction

During the last two decades much effort has been devoted to build computational architectures of musical sequence learning [11, 8]. The result of this research in musical intelligence has often inspired music psychology experiments. One example is the Continuator [22] that has been used to study childhood flow-experience [3]. Moreover, these systems naturally lead to the philosophical question about the nature of "style" in music. The problem has been attacked from many perspectives but the debate among musicologists remains open. Mayer [20] arrives at the conclusion that style is not only a complex concept originating from the interplay of different description levels of a musical piece, but also it is impossible to separate "style" from the social context in which the music has grown.

Many studies have been conducted for the analysis and the generation of music sequences. In particular, in [22], a MIDI-based system for real-time musical interaction was developed, yielding good jazz style music generation.

The handling of memory is a core challenge in music modeling [24]. Whereas the widely used bag-of-features approach neglects any sequential relations between musical events, common n-gram based methods for the representation of musical sequences usually set a maximal fixed length of context. This leads to exponentially growing storage needs to allow the model to account for more complex structures. A solution to this dilemma is offered by the different length Markov model [5]. This model determines the needed context length for each musical sequence individually, therefore maximizing storage economy, without the requirement of excessive storage.

Focusing on the question opened by machine listening systems from an information theory point of view suggests improvements to MIR techniques. In the bag-of-frames approach the distance between two audio signals is independent of the order of the notes. Since most of the musical content generally resides on the temporal organization of the sound material, essential information about the music is lost in the derived descriptors. In fact, the goal of musical intelligence systems is to learn music excerpts in a similar way

as the brain does in the first auditory scene analysis process [17]. Thus, these systems indirectly define an operative notion of style.

From a statistical point of view, "style" can be defined as a source of symbols [26]. Equivalently, we can say that, in this context, understanding the style means to find a way to compress[1]the message [29]. Another approach is in the framework of information dynamics (see [2]) in which the analysis of music is related to music cognition. Hazan et al. [14] build a system for generation of musical expectation that operates on music in audio data format. The auditory front-end segments the musical stream and extracts both timbre and time description. In an initial bootstrap phase, an unsupervised clustering process builds up and maintains a set of different sound classes. The resulting sequence of symbols is then processed by a multi-scale technique based on n-grams. Model selection is performed during a bootstrap phase via the Akaike information criterion. The method by Marxer and Purwins [19] consists of a conceptual clustering algorithm coupled with a modified hierarchical N-gram. The main flow of the system can be summarized as follows: 1) segmentation by transient detection, 2) timbre representation of each segment by Mel-cepstrum coefficients, 3) discretization by conceptual clustering, yielding a number of different sound classes (e.g. instruments) that can incrementally grow or shrink depending on the context resulting in a discrete sequence of sound events, 4) extraction of statistical regularities using hierarchical N-grams, 5) prediction of continuation, and 6) sonification.

Employment of machine learning techniques in generating musical events is crucial to achieve flexibility with respect to different musical contexts. In its architecture, our system is inspired by cognitive principles. In addition, it can be used as a validator for many of the results in music analysis in the way that the quality of the synthesis reveals if the analysis methods use to generate the synthesis have been adequate.

Part of this thesis is based on the work published in [18]. First, we define the system design and the interaction of its parts. Starting from low-level descriptors, we translate them into a "fuzzy score representation", where two sounds can either be discretized yielding the same symbol or yielding different symbols according to which level of interpretation is chosen (Chapter 1). Then we perform skeleton subsequence extraction and tempo de-

[1]This means to find a concise representation of the signal without loosing information from the original (lossless data compression). In this way we can store a message (a sequence of symbols) using less bits and then rebuild the original signal by *uncompressing* its shorter version. The complexity of the message turns out to be the key concept that determines the compression ratio (the ratio between the bits occupied by the original message and the ones occupied by the compressed message). The Lempel-Ziv algorithm is an example of such a compressor.

tection to align the score to a grid. At the end, we get a homogeneous sequence in time on which we perform the prediction. For the generation of new sequences we reorder the parts of the score, respecting the statistical properties of the sequence while at the same time maintaining the metrical structure (Chapter 2). In Chapter 3, we give a descriptive evaluation of the generated result.

Contents

Introduction **iii**

1 Unsupervised Sound Analysis Method **1**
1.1 Segmentation 2
1.2 Symbolization 2
1.3 Level Selection 7
1.4 Beat detection 10

2 Statistical Model Learning **17**
2.1 Markov Chains 18
2.2 Variable Length Markov Chains 19
2.3 Tree representation of VLMCs 21
2.4 Application to the system 22
2.5 Generation Strategies 24

3 Evaluation of the System **27**
3.1 The ENST annotations 28
3.2 Segmentation using annotations 29
3.3 Onset detection 30
3.4 "Masking" Phenomena in Timbre Clustering 32
3.5 Expert Questionnaire 35
3.5.1 Recombination Value 37
3.6 Descriptive Evaluation by Experts 37

Conclusions **41**
3.7 Future Works and Applications 42
3.7.1 Insights into the Nature of Music 42
3.7.2 Validation of perceptual models 43

3.7.3 Context aware descriptors and audio indexation 43
3.7.4 Application to Real-Time interaction systems 45

A Evaluation Tables **49**

B Software and Videos **53**

Bibliography **53**

List of Figures

1.1 General architecture of the system. 2

1.2 General idea of the analysis. **A.** Onset detection (energy attack) **B.** Feature extraction (MFCCs) **C.** Clustering of features (Single Linkage) **D.** Multilevel representation (multiple thresholds) 3

1.3 A tree representation of the similarity relationship between events (*top*) of an audio percussion sequence (*bottom*). The threshold value chosen here leads to a particular cluster configuration. Each cluster with more than one instance is indicated by a colored subtree. The events in the audio sequence are marked in the colors of the clusters they belong to. The height of each node is the distance (according to the single linkage criterion) between its two child nodes. Each of the leaf nodes on the bottom of the graph corresponds to an event. 5

1.4 A continuous audio signal (*top*) is discretized via clustering yielding a sequence of symbols (*bottom*). The numbers inside the colored triangles denote the cluster index of the event, related to the type of sound, i.e. bass drum, hi-hat, or snare. 6

1.5 A sequence is displayed in a multi-level representation. Each color represents a unique symbol, a non-trivial cluster, whereas the singletons not belonging to a non-trivial cluster are drawn in white. Note how the number of different colors increases from top to bottom, indicating that the sounds are represented in greater refinement by a larger number of clusters. 7

1.6 The procedure applied for computing the regularity value of an onset sequence (*top*) is outlined. *Middle:* the histogram of the complete IOI between onsets. *Bottom:* the autocorrelation of the histogram is shown for a subrange of IOI with relevant peaks marked. 9

1.7 Sequence regularity for a range of cluster distance thresholds (x-axis). An ENST audio excerpt was used for the analysis. The regularity reaches its maximum value in a central position. Towards the right, regularity increases and then remains constant. The selected peaks are marked with red crosses implying a list of cluster distance threshold values. 10
1.8 A skeleton sequence is represented in a timeline. *Below*, some possible alignments of the sequences are given based on the measure duration provided by the Dixon method. Each phase interpretation catches some onsets (represented with its own graphical marker) and discards some others. The phase that allows to catch more onsets (the filled red crosses) is selected and the remaining onset are removed from the skeleton grid. . . 11
1.9 The event sequence derived from a segmentation by onset detection is indicated by triangles. The vertical lines show the division of the sequence into blocks of homogeneous tempo. The red solid lines represent the beat position (as obtained by the skeleton subsequence). The other black lines (either dashed if aligned to a detected onset or dotted if no close onset is found) represent the subdivisions of the measure into four blocks. 14
1.10 From the onset score to a grid-aligned score. An example. 15

2.1 Context tree of the function $c(\cdot)$ in the example. 21
2.2 Context tree built from the analysis of the sequences {A B C D} and {A B B C}. 23

3.1 Interconnection of different fields of research with Machine Listening. . . 28
3.2 A detail of the audio with the worst onset detection F-measure. Only the recall is the problem, since the precision is 100%. The problem is clearly due to the fact that the non-detected onsets are hidden by decay tails. Listening to the sound I also noticed that I can not perceive those sounds as onsets but as an effect affecting the detected onsets. The sound is the file 099_phrase_country_simple_slow_brushes. 31
3.3 A detail of the audio with the worst detection precision. Here the sound is wrongly segmented and the generation is messy. The sound is the file 052_phrase_afro_simple_fast_sticks. 32
3.4 Evaluation questionnaire. 36
3.5 This figure shows a schematic idea of which informations a context aware descriptor should carry about the audio. 45

3.6 This figure shows an idea of which are the core features that should be compared in order to produce a similarity distance. 46
3.7 Music interaction system architecture. There is the possibility to make it interact with gestures. All the possible stage where the gesture can have a role have been linked with dashed line. 48

B.1 Interface implemented in Qt4 where is possible to record percussive audio file, analyze them and generate continuations of the desired duration. . . . 54
B.2 Frame capture from a video representing the generated output of the system. 54

Chapter 1

Unsupervised Sound Analysis Method

The system architecture consists of the following processing stages (cf. Figure 1.1 on the following page):

- Segmentation
- Symbolization
  - Feature extraction
  - Feature clustering
  - Sequence structure analysis
  - Temporal alignment
- Generation of audio
  - Adaptive cluster level determination

The audio in the input is been segmented by the onset detection algorithm producing an ordered list of audio segments of different length. Through a process of symbolization this is converted into a multilevel representation of events of the same length. The statistical analysis is then performed to the events producing a Markov Model that is used to generate a re-shuffled version of the original.
I will focus on the statistical part on the next chapter with detailed explanation. In this chapter I am going to describe the remarkable process of symbolization that was developed (the idea is explained visually in Fig. 1.2). The process is described in details in the following sections.

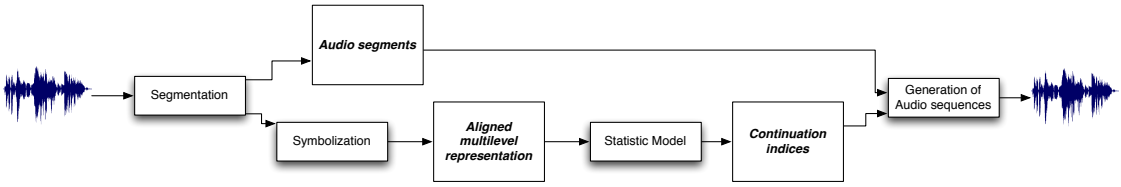

Figure 1.1: General architecture of the system.

1.1 Segmentation

First, the audio input signal is analyzed by an onset detector that segments the audio file into a sequence of musical *events*. Each event is characterized by its position in time (onset) and an *audio segment*, the audio signal starting at the onset position and ending at the following contiguous onset. In the further processing, these events will serve two purposes. On one side, the events are stored as an indexed sequence of audio fragments which will be used for the re-synthesis in the end. On the other side, these events will be compared with each other to generate a reduced score-like representation of the percussion patterns to base a tempo analysis on (cf. Fig. 1.1 and Sec. 1.2).
We used the onset detector implemented in the MIR toolbox [16] that is based only on the energy envelope, which proves to be sufficient for our purpose of analyzing percussion sounds.

1.2 Symbolization

We will employ segmentation and clustering in order to transform the audio signal into a discrete sequence of symbols (as shown in Fig. 1.4 on page 6), thereby facilitating statistical analysis. However, some considerations should be made.
As we are not restricting the problem to a monophonic percussion sequence, non-trivial problems arise when one wants to translate a sequence of events into a meaningful symbolic sequence. One would like to decide whether or not two sounds have been played by the same percussion instrument (e.g. snare, bass drum, open hi hat...) and, more specifically, if two segments *contain* the same sound in case of polyphony. With a similarity distance we can derive a value representing the similarity between two sounds but when two sounds are played simultaneously a different sound may be created. Thus, a sequence could exist that allows for multiple interpretations since the system is not able to determine whether a segment contains one or more sounds played synchronously. A way

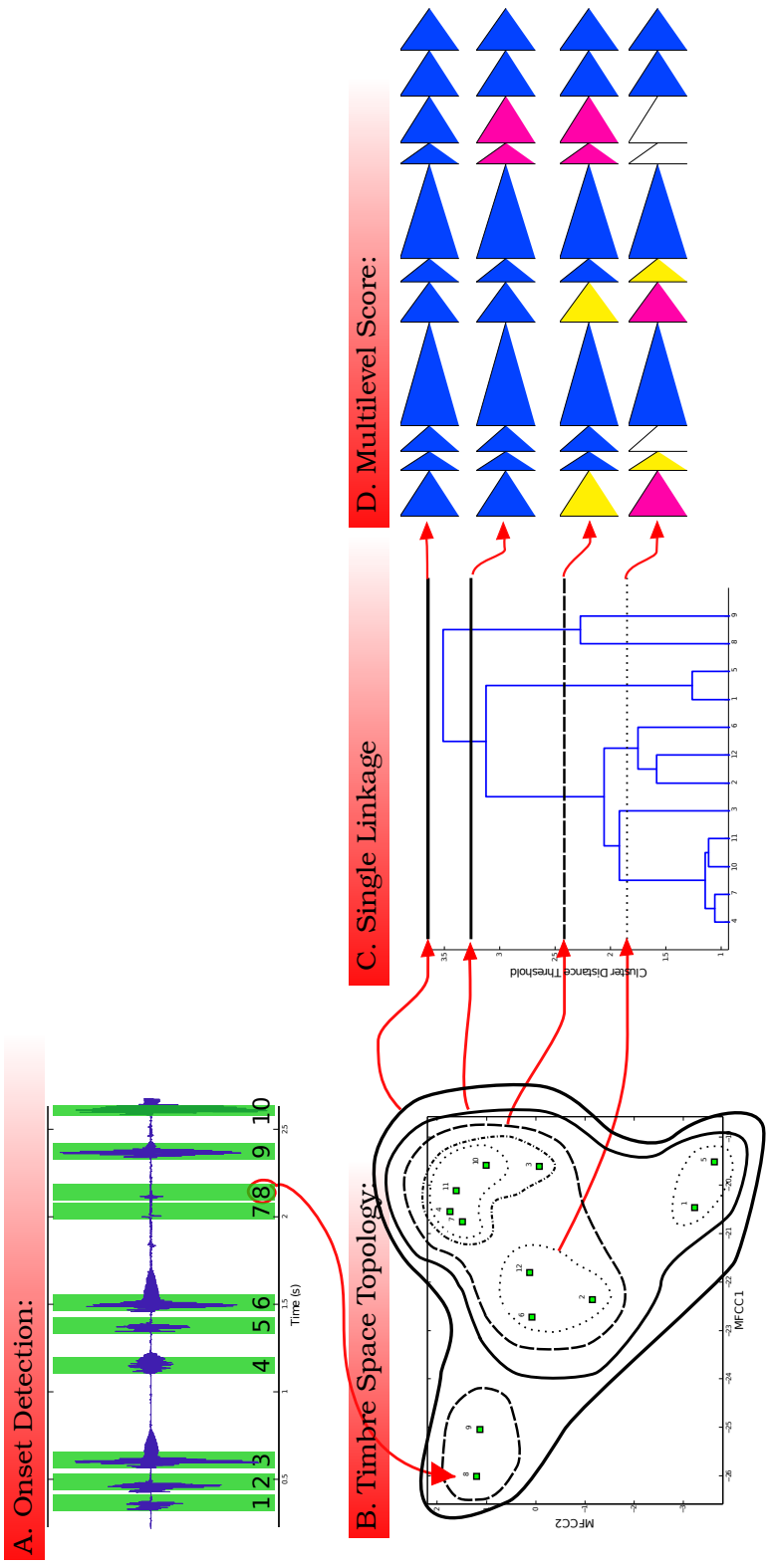

Figure 1.2: General idea of the analysis. **A.** Onset detection (energy attack) **B.** Feature extraction (MFCCs) **C.** Clustering of features (Single Linkage) **D.** Multilevel representation (multiple thresholds)

to avoid this problem directly and to still get a useful representation is to use a *fuzzy representation* of the sequence. If we listen to each segment very detailedly, every segment may sound different. If we listen very coarsely, they may all sound the same. Only listening with an intermediate level of refinement yields a reasonable differentiation in which we recognize the reoccurrence of particular percussive instruments and on which we can perceive meaningful musical structure. Therefore, we propose to maintain different levels of clustering refinement simultaneously and then select the level on which we encounter the most regular non-trivial patterns. In the sequel, we will pursue an implementation of this idea and describe the process in more detail.

Feature Extraction

We have chosen to define the *salient part* of the event as the first 200 ms after the onset position. This duration value is a compromise between capturing enough information about the attack for representing the sound reliably and still avoiding irrelevant parts at the end of the segment which may be due to pauses or interfering other instruments. In the case that the segment is shorter than 200 ms, we use the entire segment for the extraction of the feature vector. Across the salient part of the event we calculate the Mel Frequency Cepstral Coefficient (MFCC) vector frame-by-frame. Over all MFCCs of the salient event part, we take the weighted mean, weighted by the RMS energy of each frame. The frame rate is 100 frame for second, the FFT size is 512 samples and the window size 256.

Sound Clustering

At this processing stage, each event is characterized by a 13-dimensional vector (and the onset time). Events can thus be seen as points in a 13-dimensional space in which a topology is induced by the Euclidean distance.
We used the *single linkage* algorithm to discover event clusters in this space (cf. [12] for details). This algorithm recursively performs clustering in a bottom-up manner. Points are grouped into clusters. Then clusters are merged with additional points and clusters are merged with clusters into super clusters. The distance between two clusters is defined as the shortest distance between two points, each in a different cluster, yielding a binary tree representation of the point similarities (cf. Fig. 1.3 on the next page). The leaf nodes correspond to single events. Each node of the tree occurs at a certain height, representing the distance between the two child nodes. Figure 1.3 on the facing page (top) shows an example of a clustering tree of the onset events of a sound sequence.

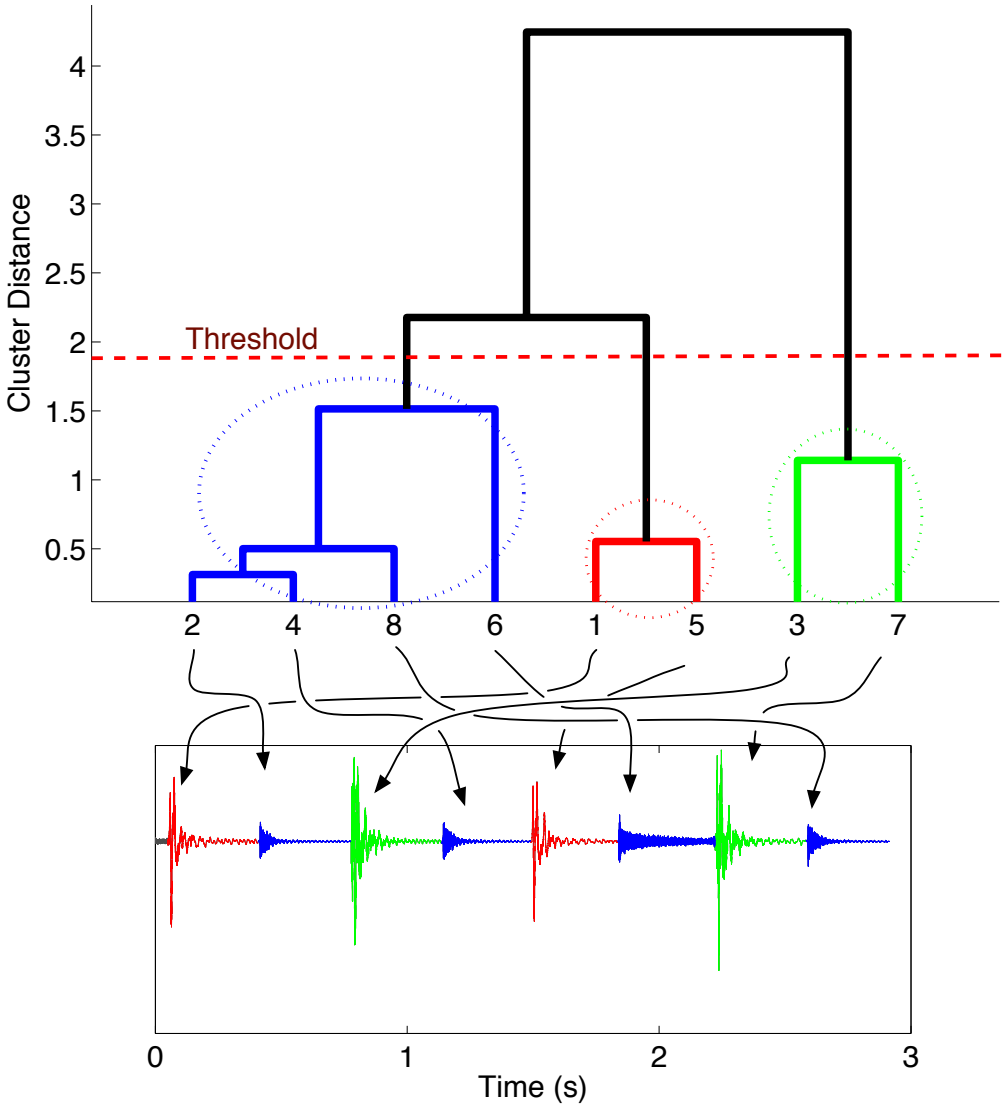

Figure 1.3: A tree representation of the similarity relationship between events (*top*) of an audio percussion sequence (*bottom*). The threshold value chosen here leads to a particular cluster configuration. Each cluster with more than one instance is indicated by a colored subtree. The events in the audio sequence are marked in the colors of the clusters they belong to. The height of each node is the distance (according to the single linkage criterion) between its two child nodes. Each of the leaf nodes on the bottom of the graph corresponds to an event.

The height threshold controls the (number of) clusters. Clusters are generated with inter-cluster distances higher than the height threshold. Two thresholds lead to the same cluster configuration if and only if their values are both within the range delimited by the previous lower node and the next upper node in the tree. It is therefore evident that by changing the height threshold, we can get as many different cluster configurations as the number of events we have in the sequence. Each cluster configuration leads to a different symbol alphabet size and therefore to a different symbol sequence representing the original audio file. We will refer to those sequences as *representation levels* or simply *levels*. These levels are implicitly ordered. On the leaf level at the bottom of the tree we find the lowest inter-cluster distances, corresponding to a sequence with each event being encoded by a unique symbol due to weak quantization. On the root level on top of the tree we find the cluster configuration with the highest inter-cluster distances, corresponding to a sequence with all events denoted by the same symbol due to strong quantization. Given a particular level, we will refer to the events denoted by the same symbol as the *instances of that symbol*. We do not consider the implicit inheritance relationships between symbols of different levels.

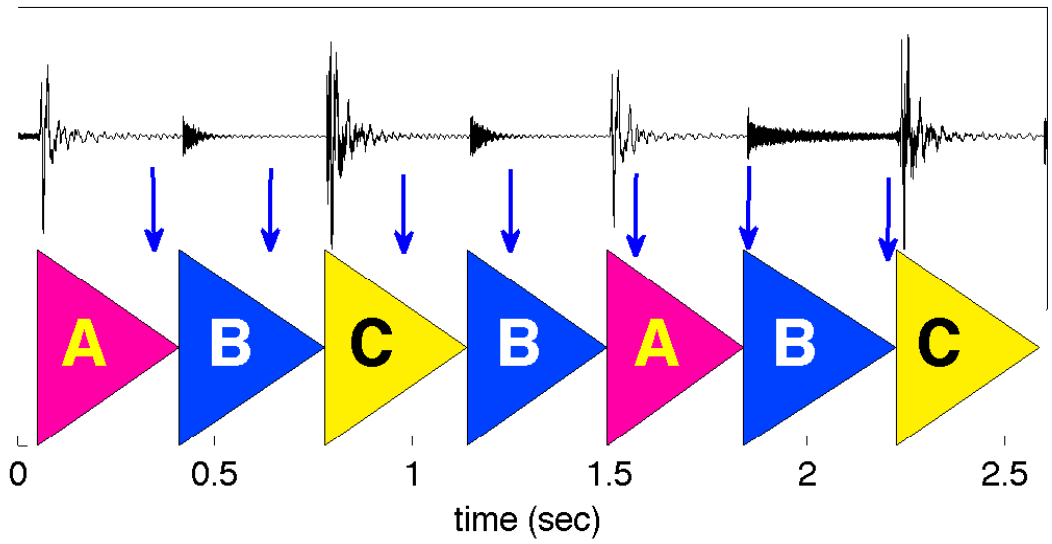

Figure 1.4: A continuous audio signal (*top*) is discretized via clustering yielding a sequence of symbols (*bottom*). The numbers inside the colored triangles denote the cluster index of the event, related to the type of sound, i.e. bass drum, hi-hat, or snare.

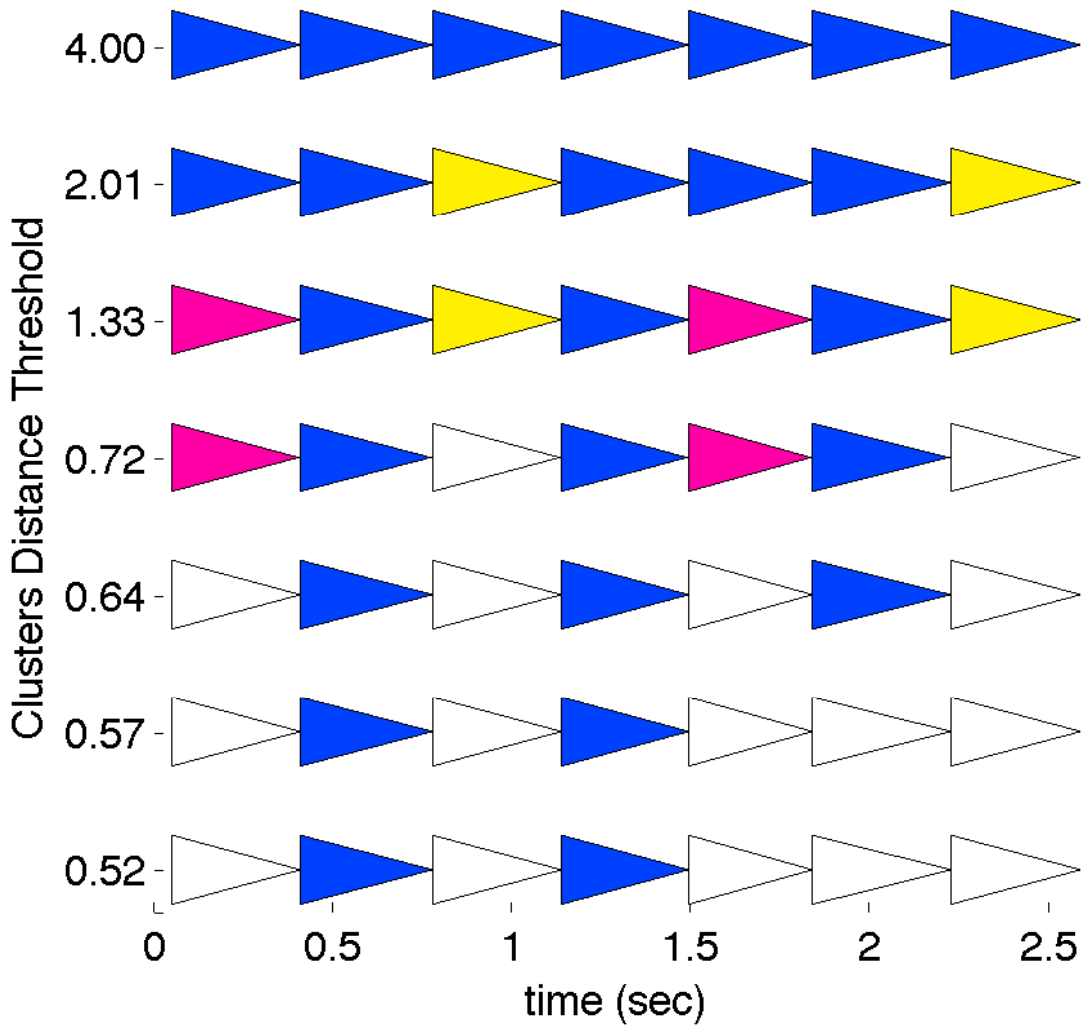

Figure 1.5: A sequence is displayed in a multi-level representation. Each color represents a unique symbol, a non-trivial cluster, whereas the singletons not belonging to a non-trivial cluster are drawn in white. Note how the number of different colors increases from top to bottom, indicating that the sounds are represented in greater refinement by a larger number of clusters.

1.3 Level Selection

Handling different representations of the same audio file in parallel enables the system to make predictions based on fine or coarse context structure, depending on the situation. As explained in the previous section, if the sequence contains n events the number of total

possible distinct levels is n (see Fig. 1.5 on the previous page). As the number of events increases, it is particularly costly to use all this levels together because the number of levels also increases linearly with the number of onsets. Moreover, as it will be clearer later, this representation will lead to over-fitted predictions of new events.
This observation leads to the necessity to only select a few levels that can be considered representative of the sequence in terms of structural regularity.
Given a particular level, let us consider a symbol σ having at least four instances but not more than 60% of the total number of events and let us call such a symbol an *appropriate symbol*. The instances of σ define a subsequence of all the events that is supposedly made of more or less similar sounds according to the degree of refinement of the level. Let us just consider the sequence of onsets given by this subsequence. This sequence can be seen as a set of points on a time line. We are interested to quantify the degree of temporal regularity of those onsets. Firstly, we compute the histogram[1] of the time differences (CIOIH) between all possible combinations of two onsets (*middle* Fig. 1.6 on the facing page). What we obtain is a sort of harmonic series of peaks that are more or less prominent according to the self-similarity of the sequence on different scales. Secondly, we compute the autocorrelation $\mathrm{ac}(t)$ (where t is the time in seconds) of the CIOIH which, in case of a regular sequence, has peaks at multiples of its tempo. Let t_{usp} be the positive time value corresponding to its upper side peak. Given the sequence of m onsets $x = (x_1, \ldots, x_m)$ we define the *regularity* of the sequence of onsets x to be:

$$\mathrm{Regularity}(x) = \frac{\mathrm{ac}(t_{usp})}{\frac{1}{t_{usp}} \int_0^{t_{usp}} \mathrm{ac}(t)dt} \log(m)$$

This definition was motivated by the observation that the higher this value the more equally the onsets are spaced in time. The logarithm of the number of onsets was multiplied by the ratio to give more importance to symbols with more instances.
Then we extended, for each level, the regularity concept to an overall *regularity of the level*. This simply corresponds to the mean of the regularities for all the appropriate symbols of the level. The regularity of the level is defined to be zero in case there is no appropriate symbol.
After the regularity value has been computed for each level, we yield the level where the maximum regularity is reached. The resulting level will be referred so as the *regular level*. We also decided to keep the levels where we have a local maximum because they generally refer to the levels where a partially regular interpretation of the sequence is achieved. In

[1] We used a discretization of 100 ms for the onset bars.

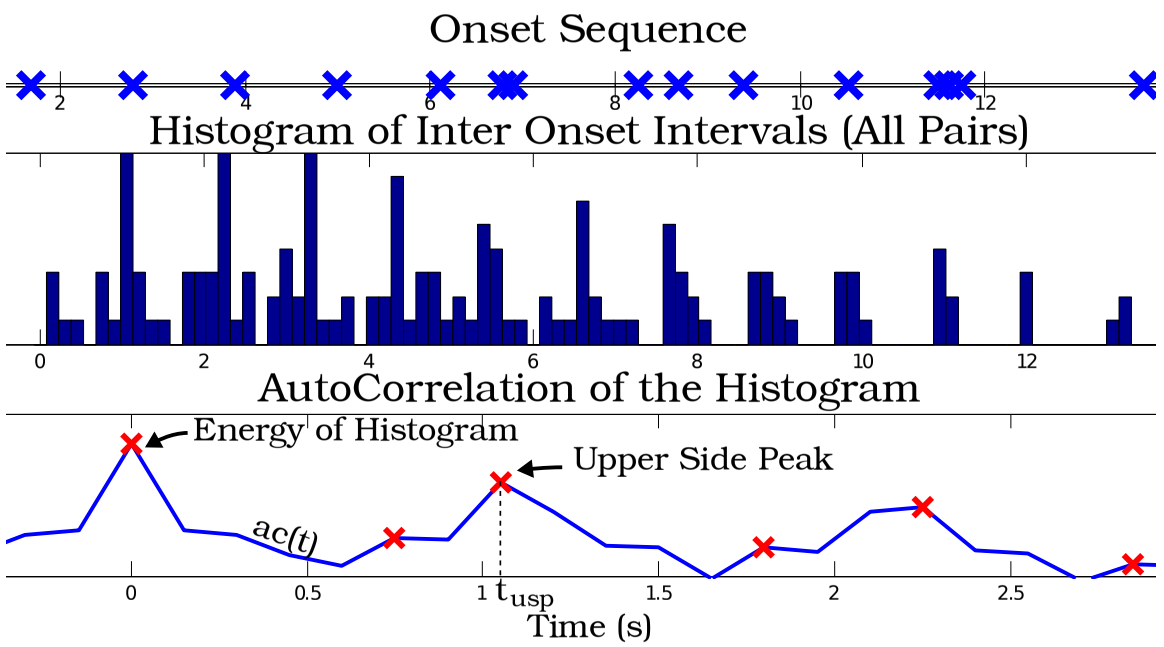

Figure 1.6: The procedure applied for computing the regularity value of an onset sequence (*top*) is outlined. *Middle:* the histogram of the complete IOI between onsets. *Bottom:* the autocorrelation of the histogram is shown for a subrange of IOI with relevant peaks marked.

the case where consecutive levels of a sequence share the same regularity only the higher one is kept. Figure 1.7 shows the regularity of the sequence for different levels.

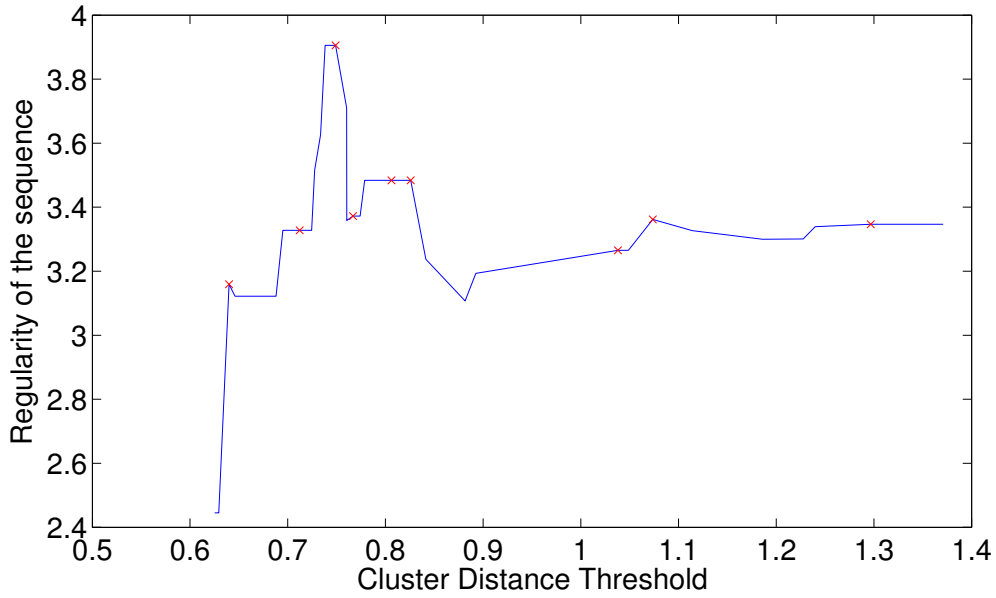

Figure 1.7: Sequence regularity for a range of cluster distance thresholds (x-axis). An ENST audio excerpt was used for the analysis. The regularity reaches its maximum value in a central position. Towards the right, regularity increases and then remains constant. The selected peaks are marked with red crosses implying a list of cluster distance threshold values.

1.4 Beat detection

In order to predict future events without breaking the metrical structure we use a tempo detection method and introduce a way to align onsets to a metrical grid.

Our starting point is the regular level that has been found with the procedure explained in the previous section. On this level we select the appropriate symbol with the highest regularity value. The subsequence that carries this symbol will be referred to as the skeleton subsequence since it is like an anchor structure to which we relate our metrical interpretation of the sequence.

Tempo Detection (Inter Beat Interval)

Once the skeleton subsequence is found, the inter beat interval is estimated with the procedure explained in [9]. The tempo is detected considering the intervals between all onset pairs of the sequence using a score voting criterion. This method tends to give higher scores to the intervals that have more instances and that share many integer ratios with other intervals.
Then the skeleton subsequence onsets are parsed in order to detect a possible alignment of the grid to the sequence. A tolerance of 6% the duration of the inter beat interval is allowed for the alignment of an onset to the grid position. We chose the interpretation that aligns the highest number of instances to grid. After discarding the onsets that are not aligned we obtain a preliminary skeleton grid. In Fig. 1.8 the procedure is visually explicated.

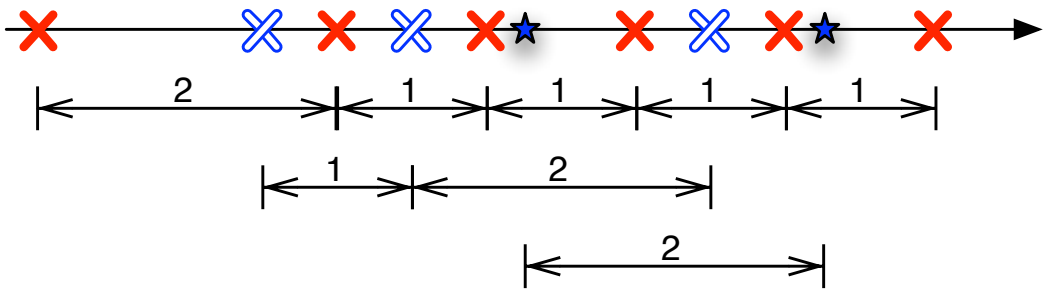

Figure 1.8: A skeleton sequence is represented in a timeline. *Below*, some possible alignments of the sequences are given based on the measure duration provided by the Dixon method. Each phase interpretation catches some onsets (represented with its own graphical marker) and discards some others. The phase that allows to catch more onsets (the filled red crosses) is selected and the remaining onset are removed from the skeleton grid.

Creation of Skeleton Grid

The preliminary skeleton grid is a sequence of onsets spaced in multiples of a constant time interval. But, as shown in the case of Fig. 1.8, it can still have some gaps (due to missing onsets). The missing onsets are, thus, detected and, in a first attempt, the system tries to align the missing onsets with one of the onsets of the entire event sequence (not only from the onsets of a certain symbol). A tolerance value of 6% determines whether

there is no onset to be aligned and, in this case, the system creates a grid bar in the expected beat position.
At the end of this completion procedure, we obtain a *skeleton grid* that will be considered to be a sequence of beats or, more generally, a sequence of events sharing the same metrical position (the same phase).
Because of the tolerance used for building such a grid it could be noticed that sometimes the effective measure duration could be slightly longer or slightly shorter. This fulfills the idea that the grid should be elastic in the sense that, up to a certain degree, it adapts to the timing of the actual sequence.
The skeleton grid catches a part of the complete list of onsets, but we would like to built a grid where most of the onsets are aligned. Thereafter, starting from the skeleton grid, the intermediate point between every two subsequent beats is found and aligned with an onset (if it exists in a tolerance region otherwise a place-holding onset is added). The procedure is recursively repeated until at least 80% of the onsets are aligned to a grid position or the number of created onsets exceeds the number of total onsets.
In Fig. 1.9 on page 14, an example is presented along with the resulting grid where the skeleton grid, its aligned, and the non-aligned subdivisions are indicated by different line markers.
Note that, for the sake of simplicity, our approach assumes that the metrical structure is binary. This causes the sequence to be eventually split erroneously. However, we will see in a ternary tempo example that this is not a limiting factor for the generation because the statistical representation somehow compensates for it even if less variable generations are achieved. A more general approach could be implemented with little modifications.
The final grid is made of *blocks* of time of almost equal duration that can contain none, one, or more onset events. It is important that the sequence given to the statistical model is almost homogeneous in time so that a certain number of blocks corresponds to a defined time duration.
We used the following rules to assign a symbol to a block (cf. Fig 1.9 on page 14):

- blocks starting on an aligned onset are denoted by the symbol of the aligned onset,
- blocks starting on a non-aligned grid position are denoted by the symbol of the previous block.

Finally, a phase value is assigned to each block describing the number of grid positions passed after the last beat position (corresponding to the metrical position of the block).

For each representation level the new representation of the sequence will be the Cartesian product of the instrument symbol and the phase.

Figure 1.9: The event sequence derived from a segmentation by onset detection is indicated by triangles. The vertical lines show the division of the sequence into blocks of homogeneous tempo. The red solid lines represent the beat position (as obtained by the skeleton subsequence). The other black lines (either dashed if aligned to a detected onset or dotted if no close onset is found) represent the subdivisions of the measure into four blocks.

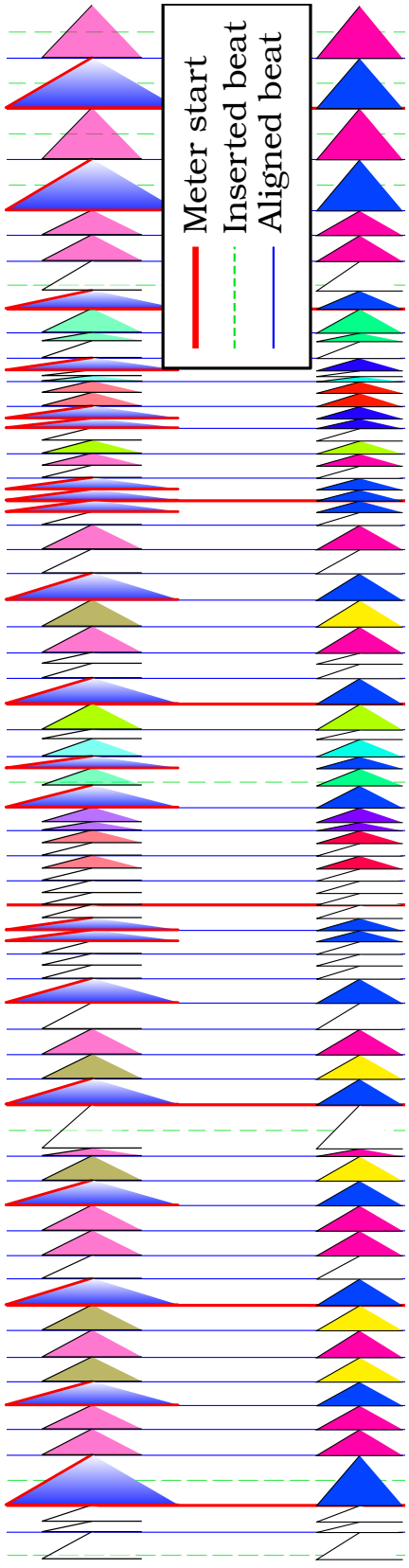

Figure 1.10: From the onset score to a grid-aligned score. An example.

Chapter 2

Statistical Model Learning

The concept of "style" in music is notably complex and often, as in other fields, not definite. Depending on the case, style can be understood looking at some details or the overall structure of the musical piece. It is almost always the case that, before taking decisions about the style, musicologists contextualize the opus by gathering informations as: author, his intent, the place where he worked, who commissioned the work and whatever might help. The statistical analysis we employ here should not require any information about the context but rely only on statistics and information theory in order to identify the style.

If we look at the problem of defining/identifying the style from the point of view of information theory, we can think of a music composition as the output of an *information source*. In this way, the problem of identifying the style is translated into the problem of identifying the information source that generated the music.

The concept of information source will intentionally be left unexplained. It can be thought as a complex (deterministic) system that outputs symbols/sounds (e.g. the brain and the body of the musician interacting with the environment). With some simplifications we will assimilate the concept of information source to concept of a *stochastic process*:

Definition 2.1. Let $(\Omega, \mathscr{F}, P)$ a probability space and $\mathscr{X}$ a finite alphabet (a set whose elements are called symbols). A *stochastic process* is a series $(X_n)_{n\geq 0}$ of random variables $X_n : \Omega \to \mathscr{X}\ \forall n \in \mathbb{N}$. Morover, the process is called *stationary* if $P(X_1 = a_1, \ldots, X_n = a_n) = P(X_{1+k} = a_1, \ldots, X_{n+k} = a_n)\ \forall a_1, \ldots, a_n \in \mathscr{X}, \forall k, n \in \mathbb{N}$.

A particular case of a stochastic process is a *Markov chain*. We will approximate stochastic processes with Markov chains.

2.1 Markov Chains

To explain a Markov chain we first have to introduce the concept of a transition nucleus:

Definition 2.2. Let $(\Omega, \mathscr{F})$ be a measurable space and $\mathscr{X}$ a finite set. We define a *discrete transition nucleus* in $\mathscr{X}$ any real function

$$N : \mathscr{X} \times \mathscr{X} \to [0, \infty]$$

such that:

$$\sum_{y \in \mathscr{X}} P(x,y) = 1 \quad \forall x \in \mathscr{X}. \tag{2.1}$$

Remark 1. Let $\mathscr{X}$ be finite set with n elements ($|\mathscr{X}| = n$) $\mathscr{X} = \{x_1, \ldots, x_n\}$. A discrete transition nucleus N on the set $\mathscr{X}$ is completely determined by the real $n \times n$ matrix $(a_{i,j})_{i,j=1,\ldots,n}$ where the generic element $a_{i,j}$ is equal to $N(x_i, x_j)$. Such a matrix is called *transition matrix* for the nucleus N and, vice versa, an $n \times n$ matrix such that the sum of the elements on each row is1 uniquely determines a transition nucleus.

Let $(\Omega, \mathscr{F}, P)$ be a probability space equipped with a *filtration*, i.e. a non decreasing sequence $\mathscr{F}_k$ of σ-algebras[1] in $\mathscr{F}$.

Definition 2.3. Let $(X_k)_{k \geq 0}$ a series of random variable with values in $\mathscr{X}$ (with the measure of the discrete space). The series is said to be an *homogeneous Markov chain* of nucleus N if it is adapted to the filtration (i.e. for all $k \geq 0$ the random variable X_k is $\mathscr{F}_k$-measurable) and moreover the following equation holds:

$$P(A, X_k = a, X_{k+1} = b) = P(A, X_k = a) N(a,b) \tag{2.2}$$

for all $k \geq 0$ and for all the possible choices for the event A in $\mathscr{F}_k$ and for all $a, b \in \mathscr{X}$.

Remark 2. If B is a subset of $\mathscr{X}$, then by posing

$$N(a,B) := \sum_{b \in B} N(a,b) \quad (\leq 1 \text{ for } 2.1) \tag{2.3}$$

we have

$$P(A, X_k = a, X_{k+1} \in B) = P(A, X_k = a) P(a,B). \tag{2.4}$$

[1] Intuitively, the elements of $\mathscr{F}_k$ represent the events of the past history up until the time k included. In other words, by knowing the history of the system up to time n included i can claim with certainty if the generic event $A \in \mathscr{F}_k$ occurred or not.

Where the latter is the more general formulation of equation 2.2. In the case the event $\{A, X_k = a\}$ has non-null probability we can rewrite equation 2.4 in terms of conditional probability, as follows:

$$P(X_{k+1} \in B | A, X_k = a) = P(a, B). \tag{2.5}$$

The logic relation 2.5 states that the position of the system at time $k+1$ conditioned on an event of type $\{A, X_k = a\}$, where A is in the past of k, it is given by $P(a, \cdot)$ and does not depend on the choice of A. To summarize: once we know the actual position, any other information on the past history does not change the distribution for the next state.

More generally, it is possible to define *non-homogeneous* Markov chains where the nucleus N in 2.2 depends on k, but for sake of simplicity I have not considered them and I will always refer to homogeneous Markov chains.
Moreover, it is possible to define Markov chains of order m where the distribution over the next state only depends on the last m positions. It is, however, possible to see a m-order Markov chain as a first order Markov chain built on an alphabet made of vectors of length m with components in $\mathscr{X}$. For example, a second order Markov chain can be translated into a first order one by substituting the series:

$$\dots a_0, a_1, a_2, \dots, a_k, \dots$$

with the following

$$\dots a_{-1}a_0, a_0a_1, a_1a_2, \dots, a_{k-1}a_k, \dots$$

and the new nucleus is defined by:

$$N(x_1x_2, y_1y_2) = \begin{cases} 0 & \text{if } x_2 \neq y_1 \\ P(X_2 = y_2 | X_0 = x_1, X_1 = x_2) & \text{otherwise} \end{cases}$$

2.2 Variable Length Markov Chains

First, we have to introduce some terms we are going to use later.

Definition 2.4. Let $\mathscr{X}$ the usual finite alphabet, we define the *set of string* over the alphabet $\mathscr{X}$ the set

$$\mathscr{X}^* := \bigcup_{k \geq 0} \mathscr{X}^k$$

whose elements are called *strings*. Note that by convention $\mathscr{X}^0 = \{\mathbf{e}\}$ where $\mathbf{e}$ is the void string.

Let $(X_t)_{t\in\mathbb{Z}}$ be a time-homogeneous Markov chain of order m with values in $\mathscr{X}$. We denote with x_i^j the following string (written in reversed order)

$$(x_j, x_{j-1}, \ldots, x_i)$$

where $i < j, i, j \in \mathbb{Z} \cup \{-\infty, +\infty\}$ and where $wu = (w_{|w|}, \ldots, w_2, w_1, u_{|u|}, \ldots, u_2, u_1)$ is the concatenation of the strings w and u.

Then, with the introduced conventions, we have:

$$\mathbb{P}[X_1 = x_1 | X_{-\infty}^0 = x_{-\infty}^0] = \mathbb{P}[X_1 = x_1 | X_{-m+1}^0 = x_{-m+1}^0], \text{for all } x_{-\infty}^1. \tag{2.6}$$

Let's introduce the concept of *Variable Length Markov chain*, which can be seen as regrouping together several states x_{-m+1}^0 in (2.6).

Definition 2.5. Let $(X_t)_{t\in\mathbb{Z}}$ be a stationary process with $X_t \in \mathscr{X}$, $|\mathscr{X}| < \infty$. Moreover, let $c : \mathscr{X}^\infty \to \mathscr{X}^\infty$ be a function such that:

$$c : x_{-\infty}^0 \mapsto x_{-\ell+1}^0, \text{where } \ell \text{ is defined by}$$
$$\ell = \ell(x_{-\infty}^0) = \min\{m; \mathbb{P}[X_1 = x_1 | X_{-\infty}^0 = x_{-\infty}^0] = \mathbb{P}[X_1 = x_1 | X_{-m+1}^0 = x_{-m+1}^0] \text{ for all } x_1 \in \mathscr{X}\},$$

and the case $\ell = 0$ holds when there the next state is independent from the context.

Then, $c(\cdot)$ is called *context function* and for all $t \in \mathbb{Z}$, $c(x_{-\infty}^{t-1})$ is called *context* for the variable x_t.

Then, the context is precisely the part of the past that influences the current value.

Definition 2.6. Let $(X_t)_{t\in\mathbb{Z}}$ a stationary process where $X_t \in \mathscr{X}$, $|\mathscr{X}| < \infty$ with corresponding context function $c(\cdot)$ as before. Let $0 \le m \le \infty$ be the smallest integer such that

$$|c(x_{-\infty}^0)| = \ell(x_{-\infty}^0) \le m \text{ for all } x_{-\infty}^0 \in \mathscr{X}.$$

Then $c(\cdot)$ is called context function of order m, and if $m < \infty$,the process $(X_t)_{t\in\mathbb{Z}}$ is called *Variable Length Markov chain* (VLMC) of order m.

I will often identify the Variable Length Markov Chain $(X_t)_{t\in\mathbb{Z}}$ with its law P_c (in the space $\mathscr{X}^{\mathbb{Z}}$). Moreover if P is a probability measure over $\mathscr{X}^{\mathbb{Z}}$ we use the following notations:

$$P(x) := \mathbb{P}_P[X_1^m = x] \quad (\forall x \in \mathscr{X}^m)$$
$$P(x|w) := \frac{P(xw)}{P(w)}.$$

2.3 Tree representation of VLMCs

Because of the homogeneity a Variable Length Markov Chain P_c is completely determined by the following transition probabilities:

$$\mathbb{P}_{P_c}[X_1 = x_1 | X_{-\infty}^0 = x_{-\infty}^0] = p(x_1|c(x_{-\infty}^0)), x_{-\infty}^1 \in \mathscr{X}^\infty.$$

The states determining such transition probabilities are given by the image of the context function $c(\cdot)$. It's convenient to represent those states in a tree form.

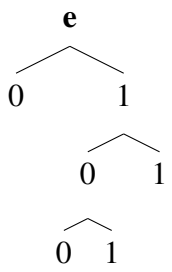

Figure 2.1: Context tree of the function $c(\cdot)$ in the example.

We consider trees with a root node on the top and whose branches grow from top to bottom. From each node a maximum number of $|\mathscr{X}|$ branches are growing. The tree is built in such a way that each value of the function $c(\cdot) : \mathscr{X}^\infty \to \bigcup_{k=0}^n \mathscr{X}^k$ is represented by a terminal node. The context $w = c(x_{-\infty}^0)$ can, in fact, be found in the tree starting from the root node and following branch given by x_0, then the sub-branch given by x_{-1} and so on until the sub-brach determined by $x_{-\ell(x_{-\infty}^0)+1}$ is reached. The resulting tree is not necessarily complete i.e. the nodes of the tree might have any number of branches equal or less then $|\mathscr{X}|$.

Example 2.1. The following context function:

$$c(x_{-\infty}^0) = \begin{cases} 0, & \text{if } x_0 = 0, \text{ and for any } x_{-\infty}^{-1} \\ 1,0,0, & \text{if } x_0 = 1, x_{-1} = 0, x_{-2} = 0, \text{ and for any } x_{-\infty}^{-3} \\ 1,0,1, & \text{if } x_0 = 1, x_{-1} = 0, x_{-2} = 1, \text{ and for any } x_{-\infty}^{-3} \\ 1,1, & \text{if } x_0 = 1, x_{-1} = 1, \text{ and for any } x_{-\infty}^{-2} \end{cases}$$

is represented by the tree of Fig. 2.3.

Now let's formalize what we have said:

Definition 2.7. Let $c(\cdot)$ be the context function of a Variable Length Markov Chain stationary. The contextual tree τ is the set:

$$\tau = \tau_c = \{w; w = c(x^0_{-\infty}), x^0_{-\infty} \in \mathscr{X}^{\infty}\} = c(\mathscr{X}^{\infty}),$$

and its elements are called *nodes*. The "topology" of such a tree (the interconnection between the nodes in τ) is given by the following rule $\omega, \tilde{\omega} \in \tau$:

$$\tilde{\omega} \text{ is child of } \omega \Longleftrightarrow \exists \sigma \in \mathscr{X} : \omega\sigma = \tilde{\omega}. \tag{2.7}$$

We automatically get the tree recursively applying the rule and after completing the nodes adding all the possible missing prefix of the strings in τ included the void prefix **e**. We, thus, obtain a tree with root node **e** and where the nodes of level i are strings in τ with i symbols.

Example 2.2. Let's consider the contex function c of example 2.1, we have in this case:

$$\tau_c = c(\{0,1\}^{\infty}) = \{0, 100, 101, 11\}.$$

Adding all the prefixes we get:

$$\tilde{\tau} = \{\mathbf{e}, 0, 1, 10, 11, 100, 101\}$$

where with the rule 2.7 we derive that 0 and 1 are sons of **e**, 10 and 11 are sons of 1, 100 and 101 are sons of 10. Thus, we exactly obtain the tree of Fig. 2.3 as we wished.

2.4 Application to the system

Now we statistically analyze the structure of the symbol sequence obtained in the last chapter. We employ variable length Markov chains (VLMC) for the statistical analysis of the sequences. This means that we will have to induce a Variable Length Markov Chain from a sequence of symbols.

In [5, 25], a general method for inferencing long sequences is described. For faster computation and because of out short examples, we use a simplified implementation as described in [22]. In fact, after deciding what is the maximal length of a context, we build the complete suffix tree for the sequence cutted at the maximal level. Each node of the tree represents a specific context that had occurred in the past. In addition, as explained in [22], each node carries a list of continuation indices corresponding to block indices matching the *context*.

We repeat this process for all the levels constructing a suffix tree based on the sequence of that level.
For audio, a different approach has been applied in [10]. This method does not require an event-wise symbolic representation since it employs the factor oracle algorithm. VLMC has not been applied to audio before, because of the absence of an event-wise symbolic representation we presented above.

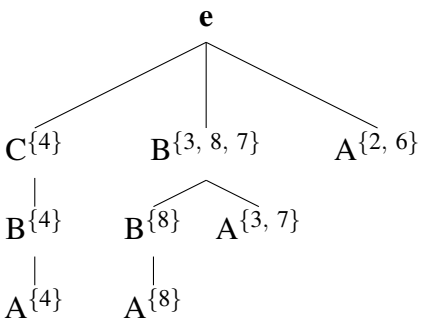

Figure 2.2: Context tree built from the analysis of the sequences {A B C D} and {A B B C}.

Suppose that we have constructed the suffix tree of figure 2.2 of one level and we want to continue the sequence {A B}. We use the suffix tree to find the longest context possible. We start from the root node an go through the tree and considering the sequence of symbols of our context in reverse order (from the most recent state to the first). We find as continuation indexes the following indices:

$$\text{Continuation\_List}(\ \{\text{A B}\}\ ) = \{3, 7\}.$$

Those indices correspond to the symbols {C, B} respectively. To generate the continuation we choose randomly between one of the two. Let's suppose that we get B. In this case we will start again with the new sequence {A B B} for which we have:

$$\text{Continuation\_List}(\ \{\text{A B B}\}\ ) = \{8\}.$$

Now we have to choose the only symbol in index 8 corresponding to C. We thus have now the sequence {A B B C} that is a new sequence never occurred in the examples provided. Going on of another step we have to continue using the longest context which is B C}. We can only choose the index 4 as continuation. So we have obtained the sequence A B B C D} that do not have any continuation. In this case we have to choose a continuing symbol using a blank context **e** and that means to choose randomly between

all of the indexes. We thus have here a "discontinuity" of the generation. To smooth out those kind of continuity we will then use other the levels of representation using the generation strategy explained in the next section.

2.5 Generation Strategies

If we fix a particular level the continuation indices are drawn according to a posterior probability distribution determined by the longest context found. But which level should be chosen? Depending on the sequence, it could be better to do predictions based either on a coarse or a fine level but it is not clear which one should be preferred. First, we selected the lower level at which a context of at least $\hat{l}$ existed (for a predetermined fixed $\hat{l}$, usually $\hat{l}$ equal 3 or 4). This works quite good for many examples. But in some cases a context of that length does not exist and the system often reaches the higher level where too many symbols are provided inducing too random generations. On the other side, it occurs very often that the lower level is made of singleton clusters that have only one instance. In this case, a long context is found in the lower level but since a particular symbol sequence only occurs once in the whole original segment the system replicates the audio in the same order as the original. This behavior often leads to the exact reproduction of the original until reaching its end and then a jump at random to another block in the original sequence. In order to increase recombination of blocks and still provide good continuation we employ some heuristics taking into account multiple levels for the prediction. We set p to be a recombination value between 0 and 1. We also need to preprocess the block sequence to prevent arriving at the end of the sequence without any musically meaningful continuation. For this purpose, before learning the sequence, we remove the last blocks until the remaining sequence ends with a context of at least length two. We make use of the following heuristics to generate the continuation in each step:

- Set a maximal context length $\hat{l}$ and compute the list of indices for each level using the appropriate suffix tree. Store the achieved length of the context for each level.
- Count the number of indices provided by each level. Select only the levels that provide less than 75% the total number of blocks.
- Among these level candidates, select only the ones that have the longest context.
- Merge all the continuation indices across the selected levels and remove the trivial continuation (the next onset).

- In case there is no level providing such a context and the current block is not the last, use the next block as a continuation.
- Otherwise, decide randomly with probability p whether to select the next block or rather to generate the actual continuation by selecting randomly between the merged indices.

Chapter 3

Evaluation of the System

Since the system is made of several stages for analyzing the sound and converting it to an event-wise representation, the evaluation of the system is non-trivial. Even though, in some simple cases, the analysis of the sound produces a meaningful transcription particularly fitted for the statistical analysis. In most of the cases, we cannot find a clear one-to-one mapping of the score to the multilevel representation. The information about the content of the sequence is in fact, somehow, distributed across different levels. Moreover, a one-to-one mapping of the score may not even be desirable. In fact, in the audio there is some information that is lost in the score, such as accents, dynamics and the interaction of sound decays. The result of all those factors leads to a "perceived score" which is more representative than the actual score but that we can not estimate.

Finally, the question is, what are we evaluating? We need to be very clear about *what the system is made for* to answer this question. But that is not that obvious. What I have developed is a prototype which demonstrates the ability to understand the style of a percussive sequence by generating similar sequences. The understand-and-generate paradigm is the central framework of Machine Listening even though it is very difficult to evaluate directly.

In the literature, indirect ways to evaluate such a system are proposed. The Continuator is evaluated from a human-computer interaction point of view. While in [21] a subjective evaluation is performed on the system showing the impressions/reactions of both professionals and children, in [3] a more systematic evaluation is performed only with children based on the Flow Theory by Mihály Csíkszentmihályi. In [7] we find an application of statistically based analysis to unsupervised music clustering. This approach is based on information distance and a similar idea could lead to an indirect evaluation of the system in the field of MIR. Then, going back to the concept of "perceived score" this may

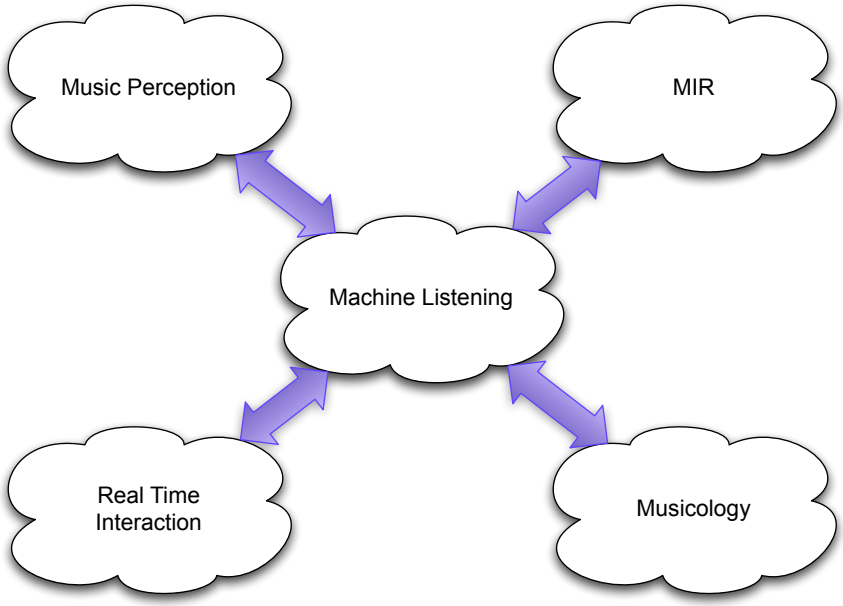

Figure 3.1: Interconnection of different fields of research with Machine Listening.

be evaluated in the field of music perception/cognition with lengthy experiments on test subjects. Finally, an interesting link is found with musicology for the concept of musical style. In our case the idea is that the style of the generation should be the same as the original.

Unluckily, the time frame did not permit me to make all those experiments. Nevertheless, I moved some steps toward the evaluation of different aspects of the system through the use of an annotated dataset (the ENST), a comparison of the produced clustering with the annotation, and an evaluation by two professional percussionists. My evaluation of the Machine Listening system do not require to embed the system into another framework and it is thus more direct. It is also adapted to the specific features of the system so that from the evaluation results some improvements ideas could emerge.

3.1 The ENST annotations

In order to start with a musically meaningful dataset of percussions sequences I used the ENST annotated dataset that has been released freely for scientific research. In [13] it is explained how the drum-kit was recorded while, at the same time, annotating the onset

of each percussion sound. See the Appendix for the list of files I used for the evaluation. The audio files come along with a text file indicating for each onset event:

- the *time* (in seconds) when the onset occured, and
- an *onset label* indicating the instrument that had been hit.

For example, this is the beginning of an annotation of a simple disco phrase:

```
0.000 cr1
0.006 bd
0.367 ohh
0.762 sd
0.770 bd
0.770 chh
1.137 ohh
1.484 chh
1.496 bd
...
```

3.2 Segmentation using annotations

The representation is thus too small-grained to have a picture of what is happening in musical terms. It is possible to see that there are some sounds played at the same time since the difference is less than a millisecond. In actual facts, the dataset is polyphonic and there can be two or more percussion sounds played at the same time.
To deal with non-monophonic percussion sequence is not a great obstacle to the project since we don't need to extract a ground-truth score in order to make predictions. Anyway, to use the dataset we have to make a couple of observations. Firstly, in the case of superimposed sounds, the feature extraction should be performed over the mix of these sounds outputting features that can be completely different from each of the single sounds. Moreover, we have to take into account that when we will use an onset detector the superimposed sounds will be represented by a unique onset. Thus, two or more sources played simultaneously should be thought of, in a first attempt, as a completely new sound.
Using the `linkage` function from the statistic library in Matlab (with a threshold of 100 ms) I created the joint annotations that, for the same example as before, looks like this:

```
0.000 bd_cr1
0.367 ohh
0.762 bd_chh_sd
1.137 ohh
1.484 bd_chh
1.860 chh
2.236 bd_chh_sd
2.555 bd_ohh
...
```

Every onset event can now be labeled with multiple percussions and the onset time is set to the one of the earlier transient.
With the new onsets that I generated it has been possible to extract segments of samples from the audio sounds. I have performed the same analysis as described in Chapter 1 using the annotated onsets. In this way, I was able to skip onset errors that may let the system classify in a different way.

3.3 Onset detection

With the joint annotations obtained I also made an evaluation of the onset detection algorithm used. This evaluation was mainly performed to see if the use of the onset detection gives reasonable results for the system. In fact, as the onset detection is used as a front-end for the segmentation, it is very important that it performs correctly.
Once again the results of the test are controversial. I computed the precision the recall[1] and the F-measure on 48 sounds (look in the table A.2 of appendix for the complete table) examples the the following mean results:

Mean Precision	Mean Recall	Mean F-Measure
88.59%	79.60%	82.72%

The F-measure resulted to be an inappropriate measure of the goodness of the onset detection for the system. In fact, for the system to work correctly, we need a good precision but the recall is relatively less important. The worst F-measure appears to be the sound of Fig. 3.2 where the F-measure is 65.57% with a precision of 100% and a recall of 48.78%. Basically one onset out of two is not detected because it is hidden by a decay tail. Listening to the sound I also noticed that I can not perceive those sounds as onsets but as

[1] I have considered an onset as retrieved with a maximum error of 50 ms.

an effect affecting the detected onsets. This case is still good for the system because the result of segmenting the sound will lead to a meaningful event representation. The generation of the continuation also worked fine.
For the system is much worst the case where the precision is low. In the case of Fig. 3.3 the precision is 59.65% the recall 80.95% and the F-measure is 68.69%. In this case the sound is unmeaningly segmented and the result of the generation is messy. Substituting the joint annotation as front-end onset detection lead to a good generation. This example shows how important is a meaningful onset detection for the system.

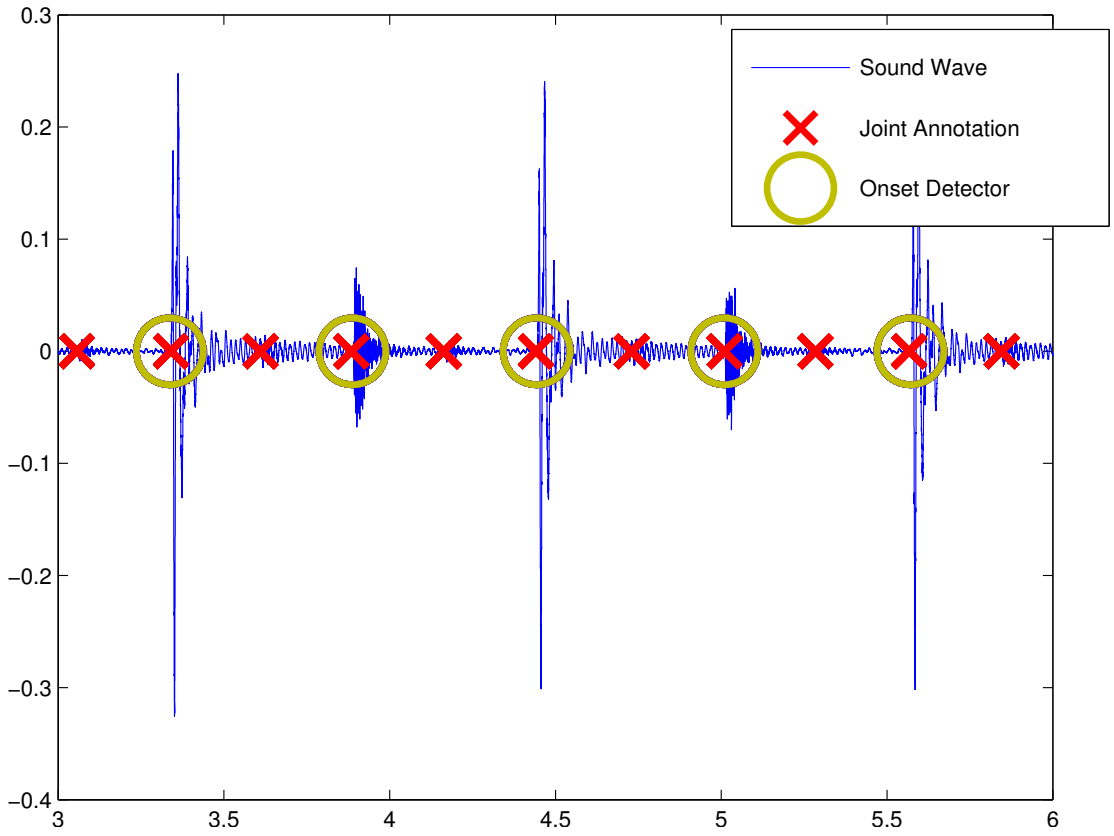

Figure 3.2: A detail of the audio with the worst onset detection F-measure. Only the recall is the problem, since the precision is 100%. The problem is clearly due to the fact that the non-detected onsets are hidden by decay tails. Listening to the sound I also noticed that I can not perceive those sounds as onsets but as an effect affecting the detected onsets. The sound is the file 099_phrase_country_simple_slow_brushes.

Thus, I found that generally for the system to work is better to have a good precision that a good recall. The system could then be improved looking for an onset detection

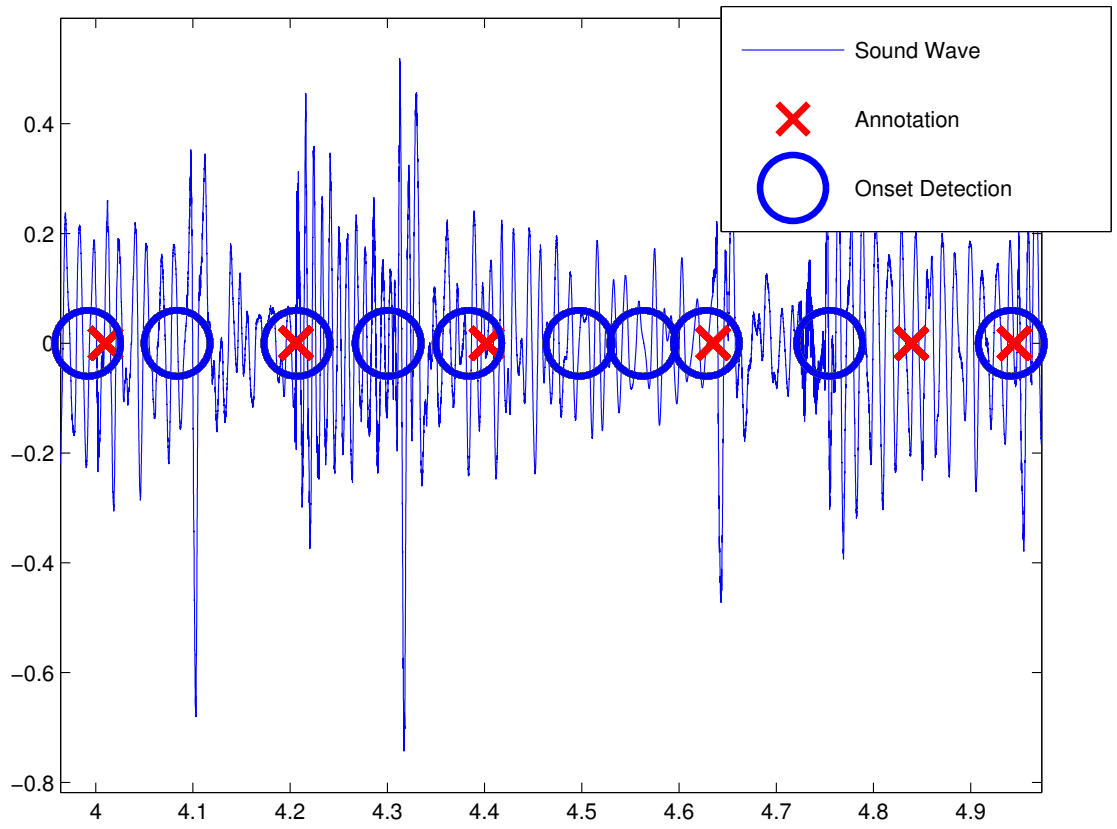

Figure 3.3: A detail of the audio with the worst detection precision. Here the sound is wrongly segmented and the generation is messy. The sound is the file 052_phrase_afro_simple_fast_sticks.

underestimating percussive events but achieving a better precision.

3.4 "Masking" Phenomena in Timbre Clustering

Another gap between the annotations and the real recording is found while doing a systematic evaluation of the clustering threshold selection. I have basically compared the symbolized version of each sound (using the onset of the annotation) from the most regular level.

Based on [19], we can define the following: Given $\mathscr{X}$, the set of the onsets and $\mathscr{P} = \{(x,x')|x,x' \in \mathscr{X}, x \neq x'\}$ the set of different onset pairs. $\mathscr{A} = \{(x,x')|a(x) = a(x')\} \subset \mathscr{P}$ the set of onsets pairs with the same annotation and $\mathscr{C} = \{(x,x')|c(x) = c(x')\} \subset \mathscr{P}$ the set of onsets pairs with in the same cluster. Then I computed the Recall R, the Precision

P and the F-measure F with the following formulas:

$$P = \frac{|\mathscr{A} \cap \mathscr{C}|}{|\mathscr{A}|} \qquad R = \frac{|\mathscr{A} \cap \mathscr{C}|}{|\mathscr{C}|} \qquad F = 2\frac{PR}{P+R}$$

Using the same 48 examples I get the following mean results:

Mean Precision	Mean Recall	Mean F-Measure
57.13%	49.09%	40.94%

This evaluation doesn't take into account that when the comparing two annotated onset, the onset `bd_chh` is different from `bd_ohh` while they sound really similar. To solve that problem I repeated the evaluation considering $a(x) = a(x')$ when they have at least one instrument in common. The precision increases significantly, meaning that when two events are in the same cluster, then it is very likely that they contain one instrument in common. This are the results with the correction:

Mean Precision	Mean Recall	Mean F-Measure
88.46%	30.47%	38.49%

Unluckily, the recall here goes down yielding a minor F-measure but this result should be commented. In fact this is due to the fact that in the results of the clustering also the accent is taken into account so that an instrument could be split into clusters characterized by different accents. Moreover the use of MFCCs produces clusters approximating a perceptual score rather than the annotated score. When listening to a percussion sequence we are not able to discriminate all the instruments composing a percussive event but we tend to focus on one instrument at time that dominates the others. Musicians use more instruments for different reasons. One may be to highlight a singular event. Another may be to create a different atmosphere, in order to differentiate between phrases. It is up to the listener to understand the meaning of a certain sound in a certain context, implicitly discriminating which instrument constitute the phrase and others are just modulating it[2].
A formalization of this idea is that there is a precedence list of instruments such that when two or more sounds are played along the higher in the list is masking the others.
In the cases where there are few sounds I was able to compute all the possible permutation of instruments and I always found out that there is an order for which the recall increases without reducing the precision. For example in the `phrase_disco_simple_slow_sticks`, using the precedence list (from most masking sound to less)

```
'sd'    'ohh'    'bd'    'chh'    'cr1'
```

Table 3.1: Transcription of an example with joint annotation, timbre clustering and masked annotation.

Joint Annotation	Cluster index	Masked Annotation
bd_cr1	4	bd
ohh	1	ohh
bd_chh_sd	2	sd
ohh	1	ohh
bd_chh	4	bd
chh	1	chh
bd_chh_sd	2	sd
bd_ohh	1	ohh
bd_chh	4	bd
chh	1	chh
bd_chh_sd	2	sd
chh	1	chh
bd_chh	4	bd
ohh	1	ohh
bd_chh_sd	2	sd
chh	1	chh
bd_chh	4	bd
chh	1	chh
bd_chh_sd	2	sd
bd_chh	1	bd
bd_chh	4	bd
bd_ohh	1	ohh
bd_chh_sd	2	sd
chh	1	chh
bd_chh	4	bd
ohh	1	ohh
bd_chh_sd	2	sd
chh	1	chh
bd_chh	4	bd
ohh	1	ohh
bd_chh_sd	2	sd
bd_chh	1	bd

we have the following annotation, clustering and masked annotation:
Here is the result using the previous correction and using the priority for this example. One can see that the first correction increases the precision significantly but reduces the recall. With the masking we get the better F-measure.

	Precision	Recall	F-Measure
Normal	46.59%	85.42%	60.29%
First Correction	66.48%	33.43%	44.49%
Masking	56.25%	86.09%	68.04%

3.5 Expert Questionnaire

I have asked to two professional percussionists to listen to 23 examples and to answer a few questions. For each example they could listen to the original percussion sequence, the most regular subsequence, and the generation. In order to keep the attention high I did not propose more than those 23 examples and I asked questions regarding the tempo of the subsequence and about the generation. Moreover, in order to keep the length of the experiment within limits, I could present only a short generated sample (30 s). But generally it may require some minutes to hear all variations proposed by the system. For this reason, I have generated 3 minutes for each example and then selected only the 30 seconds in which the generation jumps around the most in the original, i.e., as it will be explained later, the 30 second with the highest recombination value.
For each example, the musicians were asked to answer to the questionnaire of Fig. 3.4.
In the table A.1 of the appendix you can find all the answers that the percussionists provided. The main result is that for all the examples except from one the the percussionists judged that the style of the generation was the same as the original. Moreover they independently answered 100% times exactly in the same way to this question. This means statistically that there is a clear idea about the style and that the system is able somehow to capture it. Moreover it is possible to notice that the complex examples were generally judged more interesting, especially for more complex styles as funk and shuffled blues. As, from one side, it is reasonable that more complex examples capture the attention of a musicians, from the other, it is remarkable that the system is able to manage successfully the styles not only in simple cases but also with complex one. It is also remarkable to notice that in general, the generation was judged interesting. For this feature there is, however, a calibration problem. In fact, as pointed out by one of the musicians, it was not

[2]This is also another example of structural levels in music for the grouping of temporal events.

1. Overall, is the meter of the original preserved in the regular subsequence?

   A Yes, definitely

   B I can tap the beat along but it is not evident

   C No, the meter of the subsequence is not clear

2. Is the beat of the subsequence as fast as the meter of the original (mark the right answer)?

   A twice as fast

   B half as fast

   C other speed

   D tempo cannot be determined

   E same tempo

3. Is the generation respecting the style of the original?

   Yes/No

4. How musically interesting is the generation?

   Value from 1 (very uninteresting) to 5 (very interesting)

5. Do you want to comment on this example?

   Free space for writing

Figure 3.4: Evaluation questionnaire.

specified whether they had to judge the interestingness of the generation with respect to the original or in absolute terms. Even though the result is good, it is then more difficult to interpret.

Regarding the most regular subsequence tempo they also agreed 22 out of 23 that the meter of the original is preserved. But when asked to say if the tempo was the same, in integer relation with the original or different they agreed 17 times out of 23 that the tempo was half of the original. Out of the remaining 6 answer only in one case the tempo was considered different by one musician but not from the other. In this question the only bad answer for the evaluation was the case when a different tempo was detected. Less bad was the case when the tempo is twice as fast. But for the correct generation of the sequence the better answer was “same tempo” or “half as fast”. So it is possible to say that also for this question we got what we expected.

3.5.1 Recombination Value

Side by side, I tried to place the experts' answer with some objective feature of the generation and the labels of the examples. In particular, for each example, I have computed a value of *recombination* counting how many times an event segment is followed by a non-contiguous segment (i.e. how many "jumps" the generated sound contains), in the 30 second generation . There could be some variation of this value in different generations but in a long time we will expect to have the same number of jumps. However I there does not seem to be any significant correlation of this value with other features.

3.6 Descriptive Evaluation by Experts

As a descriptive evaluation I collected the comments that the percussionists attached to the questionnaire and that we shared by email.
Moreover, we asked one of the two to record two beat boxing excerpts trying to push the system to the limits of complexity and to assess critically the sequences that the system had generated from these recordings. The examples are available on the web site [1] along with some graphical animations explaining the analysis process.
Four examples were taken from the ENST database (see [13]), one from FreeSound.org and two examples were recorded with the percussionist. From the ENST database, we have selected medium/high complexity examples that we numbered according to our collection list. They are Examples no. 15, 21, 28, and 31 corresponding to the following files of the ENST database:

```
053_phrase_afro_complex_slow_sticks
072_phrase_shuffle-blues_complex_slow_sticks
079_phrase_hard-rock_complex_medium_sticks
088_phrase_waltz_simple_medium_brushes
```

From FreeSound we have selected the popular "Amen Break" loop, because of its common use and manipulations during improvisation sets.
Starting from the latter, according to the percussionist,

> «As the starting material is relatively rich the continuation is very good considering the length of the original. Especially interesting is the small looping part in 0.58s. It is very similar to what I would do as a percussionist».

It is possible to note how the metrical structure is preserved in those examples due to the introduced tempo restrictions. Moreover, an important feature is that it creates rela-

tively original variations given the short length of the learned examples. Example 28 is commented by the percussionist in the following way:

> «Very good. Meter is kept perfectly, and the "drum fills" are provided in appropriate times. A problem is that all fills are played as they appear in the song, and not extended or slightly more complex».

For Example 31, the percussionist expressed surprise for the realism of the generation and he referred to Example 15, saying:

> «The starting material is very good and rich in this case, so the continuation is rich too. Some fills are expanded and more complex which is very good, although other sequences appear exactly as they did in the original».

Referring to the beatboxing examples, he pointed out that the metrical structure is kept correctly but the beats do not vary too much in terms of accent. Then, he added:

> «Especially good is the looping of a single beat at 1.20s of the first example, although normally the looping shouldn't be repeated too much to maintain a phrase balance».

Finally, as an overall consideration, he pointed out an interesting application of the system:

> «If these continuations are used as an accompaniment, they are excellent since they, firstly, maintain a steady rhythm but at the same time evolve and, secondly, they more or less keep the time signature (i.e. strong beats usually land on the strong part of the meter, meters sound conceptually as distinct units)».

However, he also mentioned several missing features in comparison to a human percussionist solo.
From our point of view, it is worth mentioning that in Examples 21 and 31 even if the tempo is ternary the generation still preserves the metrical structure. In this case, three ternary beats constitute an event together, causing a bad representation of the sequence (a sort of swing subdivision). The statistical model is still able to select a good block position each time.
The behavior of the system depends on the correct behavior of all its parts. In particular, see [9] for a systematic evaluation of the tempo detection. Nevertheless, some examples

show that even when the computed symbolic representation of the audio does not respect directly the underlying musical sequence (e.g. the ternary tempo) the statistic model tends to generate sequences that do not break the metric structure.

Conclusions

The system that have been described and implemented is the result of many attempts to develop a machine learning system for percussive audio sequences. Due to the rhythmic nature of the sounds, I had to explicitly manage the Meter for having generations respecting the same metrical structure as the original.

This project produced many valuable results in this direction. The main contributions were achieved in the analysis of the sound. At the beginning the sound is only a continuous signal. By mean of the unsupervised method it is now possible to analyze an audio file discovering a metrical structure and a meaningful event-wise representation. The ideas that were introduced to achieve such a method have the potential to become great instruments for the automatic analysis of audio recordings. In particular, the concept of regularity of a sequence is fruitful, giving way to the ability to extract the most regular subsequence from a relatively short fragment of audio. From such a subsequence it was then possible to align a homogeneous temporal grid with a hierarchical structure, capturing the meter of the sound. This represented a first step toward the formalization of gestalt principles in machine learning. The multilevel representation is then used for the generation in other to retain any information spanning different levels of resolution.

Another important achievement is the generation strategy trading off statistical significance and clustering resolution. This leads to convincing generated sequence where the discontinuities caused by trivial statistical deductions are avoided.

The global result of all those achievements is a system that effectively generates sequences which respect the structure and the tempo of the original sound for medium/high complexity rhythmic pattern.

The questionnaire evaluation of a professional percussionist confirmed, from one side, that the metrical structure is correctly managed and, from the other hand, that the statistical representation generates meaningful sequences in the style of the original. He, in fact, noticed explicitly that the *drum fills* (short musical passages which helps to sustain the listener's attention during a break between the phrases) were managed correctly by the

system.
The critics proposed by the percussionist were directed to the lack of dynamics, agogics and of long term phrase management which we didn't consider.
Part of those feature could be achieved in the future by extending the system to the analysis of non-binary meter. To achieve musically sensible dynamics and agogics (*rallentando, accelerando, rubato...*) of the generated musical continuation for example by extrapolation [23] remains a challenge for future work.

3.7 Future Works and Applications

Resuming up, the percussionist suggested to add the following improvements:

- Allow ternary metrical subdivisions
- Integrate the tempo detection procedure with a Beat tracking to allow tempo adaptations and generations with agogics
- Detect dynamic changes and reapply them thru a "feature remapping" (see fig. 3.7) stage during the resynthesis
- Perform the prediction for each single symbol sequences (each single sound) in parallel and then mix together the results.

Those suggestions were more or less expected at the beginning of the work and are the minimum achievements that should be performed before releasing the system to the musicians.
Moreover, in the next paragraphs I propose some research directions for developing applications with the system as a core. I also underline some research questions that should be explored.

3.7.1 Insights into the Nature of Music

I expect this explorative analysis to give insights into the nature and cognition of music, especially with respect to the representation of time and saliency in music. Just building the prototype system required, in fact, to investigate some remarkable questions about music cognition and the auditory system. From one side one needs to know *what* is perceptually relevant for the auditory system. For example, we had to consider as valid only inter beat interval included in a specific range (between between 0,6 and 1,5 second) based

on insights from perceptual studies. On the other side we had an ill-posed question still worth to investigate: *where* does the musical "content" of the musical sequence resides? The question is not such innovative in the landscape of MIR research but it is also central for a vast spectrum of applications. The approach I am proposing tries to overcome problems arising in the bag-of-frames approach that have been widely used but avoided facing directly that question.

3.7.2 Validation of perceptual models

The machine listening system proposed has the potential to become a Litmus test for the goodness of the audio analysis performed. In fact, letting a musician judge a short fragment of generated music makes it very easy to discover which characteristics of the music have been lost in the process (e.g. dynamics). It is even simpler to understand if the analysis produced a wrong structure score transcription. More specifically, in my preliminary experiments, I found out that the generated sequences tend to follow two opposite tendencies. From one side, when the analysis produces an "over-fitted transcription"[3], the generated sequence doesn't break the musical stuctures but contains only few musical variations and tends to repeat constantly the same loops (in this case we say that the generation presents a *loopy* tendency). From the other side, when the transcription is over-simplified[4], the generation is very likely to break the musical structures (we say in this case that the generation is *too random*). A clear research question here is whether the understanding of the musical content of the audio can be found in an *equilibrium between loopy generation and too-random generation*. Furthermore, we may ask if there are musical genres/contexts where this doesn't hold anymore and investigate on how to deal with them.

3.7.3 Context aware descriptors and audio indexation

A question that neuroscientists are trying to answer often is what does remain in our brain after listening to a musical excerpt. Which information do we remember? What remains in the brain is also related to what reaches the attention of the listener, that condensed

[3]We have an over-fitted transcription when each event tends to be classified as unique event because of the limits of several procedures used (onset detector, descriptors...). The term "over-fitted" has a notable assonance with the homonymous concept in statistics (from which it has been taken) but is used here with a different meaning.

[4]When the richness of the music sequence is lost in the transcription

information that is "present" to the listener. Also related to this is the notion of *salience*. In the same way we can ask here, after the system has parsed and *learned* a certain audio sequence, what kind of information does it remains in the memory of the system? It is not the whole audio itself, neither it is just a sequence of low level descriptors. This information is, however, everything that the system needs to know about the sequence in order to generate interesting variations.

In Figure 3.5 a general idea of the type information a context aware descriptor should carry is presented. Take into account that, however, the figure shows only the information for one level of refinement. The starting event positions are considered as links to time positions and are stored in the descriptor together with the symbol sequence for a fast navigation in the audio. This information is what will, in fact, enable the user to *access directly the right portion of the sound* that he might be interested in.

Moreover, each symbol is related to a particular region in the feature space whose boundaries are stored in the descriptor. Finally a statistical analysis is also recorded. It is a tree structure equivalent to a Markov table over the symbols and that has been represented as a connected graph with transition probabilities over the arrows (for simplicity the figure only first-length Markov chains).

This descriptor can lead to different similarity distances that can be tuned, according to the requirements, to act like a low/high level similarity measure. Different similarity distances can, in fact, come out from several comparisons performed on the descriptors. In figure 3.6 some double stroked arrows are linking the core parts of the descriptors that should be compared. Firstly each symbol of the first descriptor is assigned to a symbol of the other sound. This could lead to multiple choices and for each one, the statistical models could be compared by aid of information theory methods (likelyhood estimation, Kulback-Leibler divergence...). Then an overall distance could be estimated choosing the reassignment that leads to the better matching between statistical models. Those comparisons will yield a "stylistic" similarity between the music sequences.

Resuming up, a context aware descriptor not only will produce a method to detect *stylistic similarities* between music excerpts but also allow for a *navigation in a collection of audio material* that has been analyzed producing generation of musical material from the combination of different excerpts. The indexation will, moreover, produce a access directly to the interested part of the audio from a large database. Finally, a further application for musicians will also be the possibility to produce rhythmic morphing. The imposition of some constraint could in fact help them to the generate stylistic/rhythmic recombinations of different musical material in the desired way.

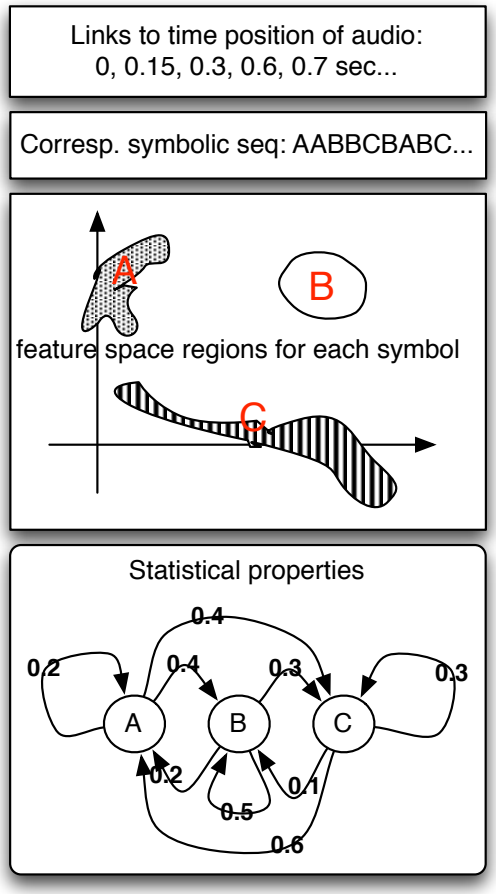

Figure 3.5: This figure shows a schematic idea of which informations a context aware descriptor should carry about the audio.

3.7.4 Application to Real-Time interaction systems

Machine listening has allowed to create musical dialogues between computers and humans. A couple of examples are the Omax system developed at IRCAM (see [4]) and the Continuator developed at Sony (see [22]).

Those systems are able to learn from musical fragments played by a performer in real-time and generate other similar sequences which respect a sort of “continuity rule”.

The system developed was inspired by the latter works and can also lead to the implementation of a real time interactions system. Figure 3.7 illustrates the general architecture of the machine learning system that was developed. The blocks marked with a question mark are possible future addition to the system in the context of real time. In fact the system

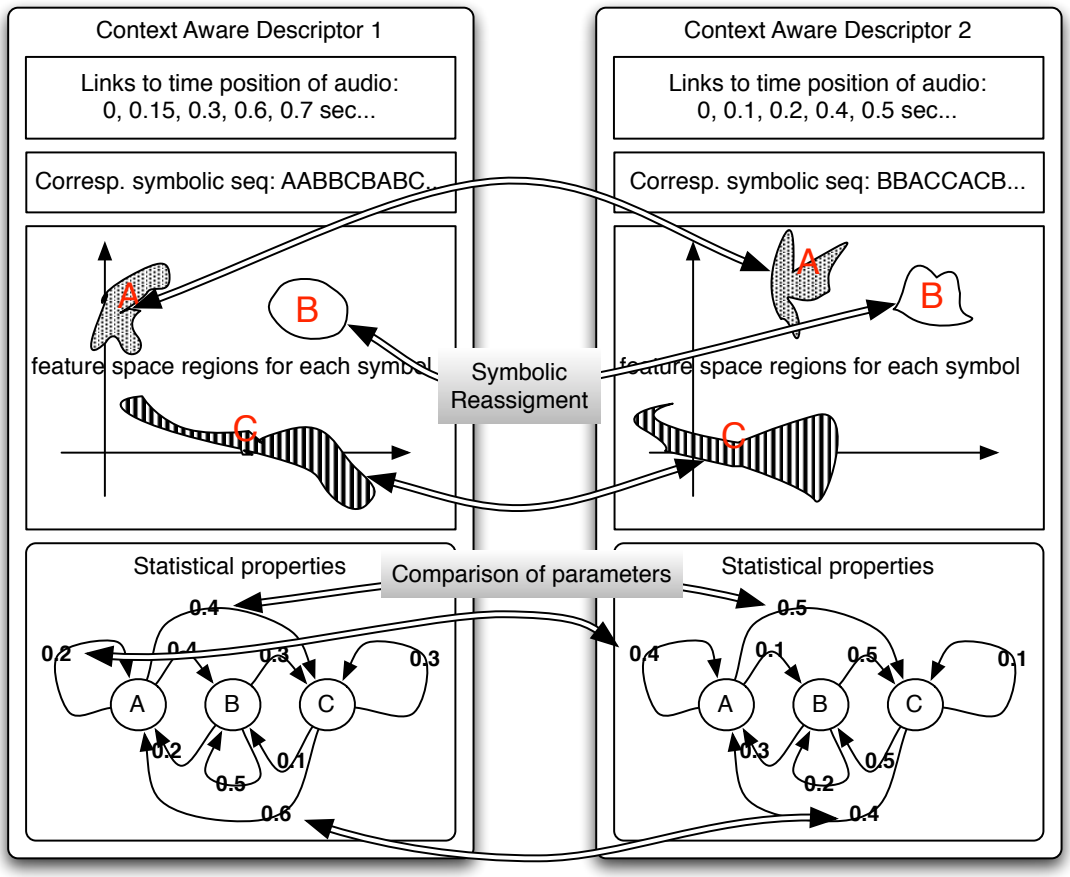

Figure 3.6: This figure shows an idea of which are the core features that should be compared in order to produce a similarity distance.

could be extended to gesture input. Another improvement would be adding goals to the system so that the generation would be driven by intelligent strategies. Finally an audio feature remapping part could be driven by the statistical model in order to apply audio transformation to the musical material during the synthesis (for example changing the loudness to manipulate the accents or the dynamics). Those improvements could, in fact, lead to a human computer interaction that would overcome the problem of the currently available interaction systems. Basically those system are able to generate music excerpts that are consistent in the short term but not in the long term. The greatest problem that arises from this kind of systems is thus managing the higher-level structure of the music that is generated.

As suggested by many studies (a starting reference could be [6, 27, 28]) performers ges-

tures are usually connected to music structures. In fact, the perception of a music performance could change significantly if the listeners were blinded or not able to see the gestures of the performer. More specifically, the gesture usually carries an anticipation of the music that is coming next or, from the the other side, force the listener to attain to a particular interpretation. It thus could also change the perceived relationship between different part of the score.

In the case of musical interaction systems it would be thus particularly indicated to have a gesture interaction between the performer and the system. The use of sensors to capture the movement of a performer/user not only is a reality now days but it is also becoming an interface paradigm. The reasons can be found both on the reduced cost this equipments and both on the popularity of accelerometers that can be found in many devices (cell phones, digital cameras...).

From a RTI point of view, the challenge is to use this low level inputs to interact with real time systems in a "fluid" manner. The potential of gesture input is, in fact, its potentiality to carry several information at the same time. For example, one gesture can be at the same time a switch that activate something (e.g. the type of gesture) and a collection of parameters carried by the way gesture itself is performed (e.g the the missing part to be completed, the instant velocity...). Moreover, since gestures are continuos inputs, if the response is in real-time the user could have the time to modulate/change the gesture continuously time by time until he achieve the desired feedback.

An example of such an application can be seen in the gesture follower developed at IRCAM (see [15] for a demo). There is a training process where the user performs gestures imitating a sound that is played in sync. After several sounds have been coupled with a gesture the user performs one of the learnt gesture. The system recognize the gesture and plays it time-stretched in order to adapt the duration of the sound to the duration of the gesture.

A problem to face would be in this case the problem of optimization. To produce a system able to work in real time it could be necessary to tune it to particular situations. At the end, the system could probably work in real time but for specific music situations (after defining e.g. a specific type of instrument, a tempo range...). The tuning procedure could be semi-automatized so that each musician could be able to perform it. In fact, there would be the possibility to use of machine learning techniques (particularly suited could be *reinforced learning*) to find the best strategies of optimizing the parameters. Eventually the system tune dynamically its parameters during the performance.

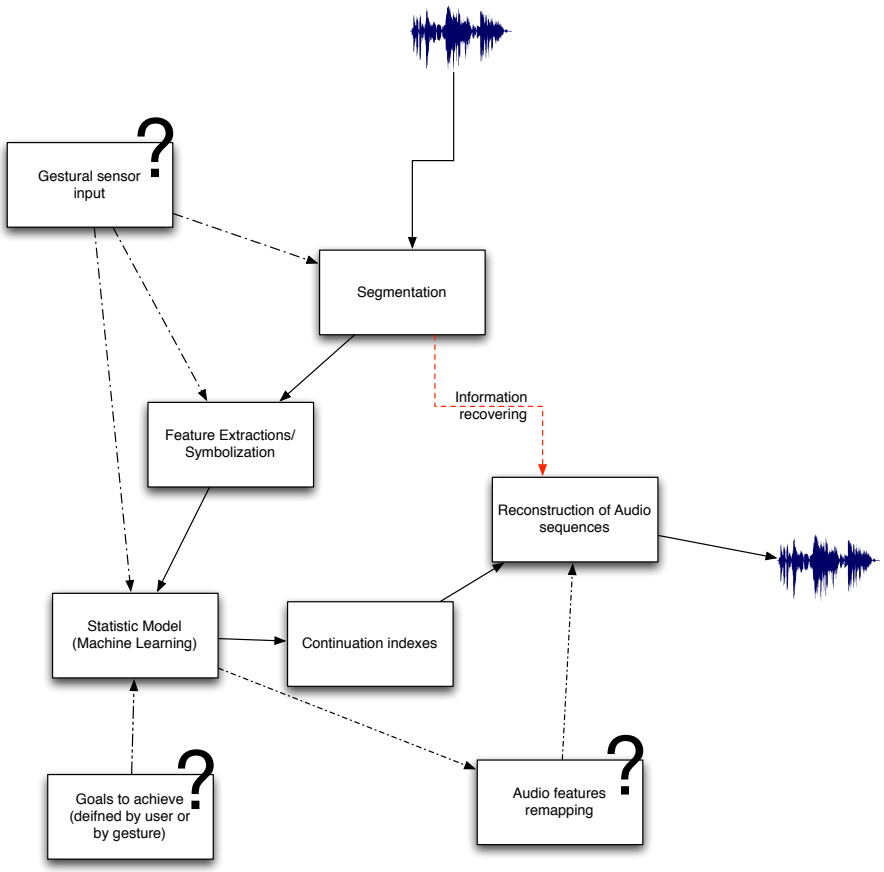

Figure 3.7: Music interaction system architecture. There is the possibility to make it interact with gestures. All the possible stage where the gesture can have a role have been linked with dashed line.

Appendix A

Evaluation Tables

Table A.1: Results of the questionnaire.

File Ground Truth Labels						Percussionist 1 answers				Percussionist 2 answers			
index	style	complexity	tempo	percussion type	number of jumps in the generation	Overall, is the meter of the original preserved in the regular subsequence?	is the beat of the subsequence as fast as the meter of the original (mark the right answer)?	is the generation respecting the style of the original?	How musically interesting is the generation?	Overall, is the meter of the original preserved in the regular subsequence?	is the beat of the subsequence as fast as the meter of the original (mark the right answer)?	is the generation respecting the style of the original?	How musically interesting is the generation?
1	disco	simple	slow	sticks	21	Yes, definitely	half as fast	Yes	4	Yes, definitely	same tempo	Yes	3
2	disco	simple	medium	sticks	32	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
3	disco	complex	slow	sticks	58	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
4	disco	complex	medium	sticks	24	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
5	disco	complex	fast	sticks	93	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
6	rock	simple	slow	sticks	65	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
7	rock	simple	medium	sticks	68	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
8	rock	simple	fast	sticks	62	Yes, definitely	half as fast	Yes	4	Yes, definitely	half as fast	Yes	4
9	rock	complex	slow	sticks	68	Yes, definitely	half as fast	Yes	4	Yes, definitely	half as fast	Yes	4
10	rock	complex	medium	sticks	92	Yes, definitely	half as fast	Yes	4	Yes, definitely	twice as fast	Yes	5
11	rock	complex	fast	sticks	52	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
18	shuffle-blues	simple	slow	sticks	51	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	4
19	shuffle-blues	simple	medium	sticks	75	Yes, definitely	half as fast	Yes	4	Yes, definitely	half as fast	Yes	5
20	shuffle-blues	simple	fast	sticks	109	Yes, definitely	half as fast	Yes	3	Yes, definitely	half as fast	Yes	3
21	shuffle-blues	complex	slow	sticks	66	Yes, definitely	half as fast	Yes	5	Yes, definitely	same tempo	Yes	5
22	shuffle-blues	complex	medium	sticks	82	Yes, definitely	half as fast	Yes	5	Yes, definitely	same tempo	Yes	5
23	shuffle-blues	complex	fast	sticks	120	Yes, definitely	half as fast	Yes	4	Yes, definitely	half as fast	Yes	4
36	funk	simple	slow	sticks	59	Yes, definitely	half as fast	Yes	4	Yes, definitely	other speed	Yes	3
37	funk	simple	medium	sticks	18	Yes, definitely	half as fast	Yes	4	Yes, definitely	half as fast	Yes	3
38	funk	simple	fast	sticks	92	Yes, definitely	half as fast	Yes	4	Yes, definitely	half as fast	Yes	4
39	funk	complex	slow	sticks	39	Yes, definitely	same tempo	No	5	I can tap the beat along but it is not evident	tempo cannot be determined	No	4
40	funk	complex	medium	sticks	61	Yes, definitely	half as fast	Yes	5	Yes, definitely	half as fast	Yes	5
41	funk	complex	fast	sticks	73	Yes, definitely	half as fast	Yes	5	Yes, definitely	half as fast	Yes	5

Table A.2: Results of the onset detector.

index	Precision	Recall	F-Measure
1	1	0,96	0,98
2	0,93	0,90	0,92
3	1	0,67	0,80
4	1	0,72	0,84
5	0,98	0,78	0,87
6	1	0,97	0,98
7	1	0,97	0,98
8	0,94	0,78	0,86
9	0,96	0,88	0,92
10	1	0,91	0,95
11	1	0,89	0,94
12	0,65	0,94	0,77
13	0,69	0,92	0,79
14	0,59	0,80	0,68
15	0,83	0,68	0,75
16	0,82	0,82	0,82
17	0,64	0,91	0,75
18	0,68	0,97	0,80
19	0,91	0,93	0,92
20	0,94	0,82	0,88
21	0,87	0,87	0,87
22	0,87	0,87	0,87
23	0,95	0,81	0,87
24	0,88	0,96	0,92
25	0,96	0,96	0,96
26	0,91	0,95	0,93
27	0,89	0,92	0,91
28	0,84	0,88	0,86
29	0,70	0,83	0,76
30	0,87	0,56	0,68
31	0,90	0,59	0,71
32	0,87	0,61	0,72
33	0,90	0,56	0,69
34	0,92	0,65	0,76
35	0,82	0,58	0,68
36	1	0,80	0,89
37	0,89	0,65	0,75
38	0,88	0,74	0,81
39	0,94	0,83	0,88
40	0,92	0,71	0,80
41	0,81	0,64	0,71
42	1	0,48	0,65
43	1	0,53	0,70
44	0,82	0,82	0,82
45	0,66	0,72	0,69
46	0,97	0,78	0,87
47	0,97	0,61	0,75
48	0,74	0,80	0,77
Mean	0,88	0,79	0,82

Appendix B

Software and Videos

From the web link [1] it is possible to listen to some of the examples with an illustrative animation or video. Let us briefly explain what can be seen in these animations. In each video, we see the original sound fragment and the generation derived from it. Each video shows an animated graphical representation (cf. B.2) where each block is represented by a triangle. The horizontal axis corresponds to the time in seconds and the vertical axis to the clustering quantization resolution. In the beginning, the original sound is played and the animation shows the discovered block representation. At each moment, the currently played block is represented by an increased colored triangle and highlighted by a vertical dashed black line. The other colored triangles highlight all the blocks from the starting point of the measure to the current block. In the second sequence, only the skeleton subsequence is played. In the last sequence, the generation is shown. The colored triangles represent the current block and the current context. The size of the colored triangles decreases monotonically from the current block backwards displaying the past time window considered by the system. The colored triangles are represented only on the levels selected by the generation strategy. The colors correspond to a symbol in a one-to-one manner.
In Figure B.1 it is shown one interface developed for the system.

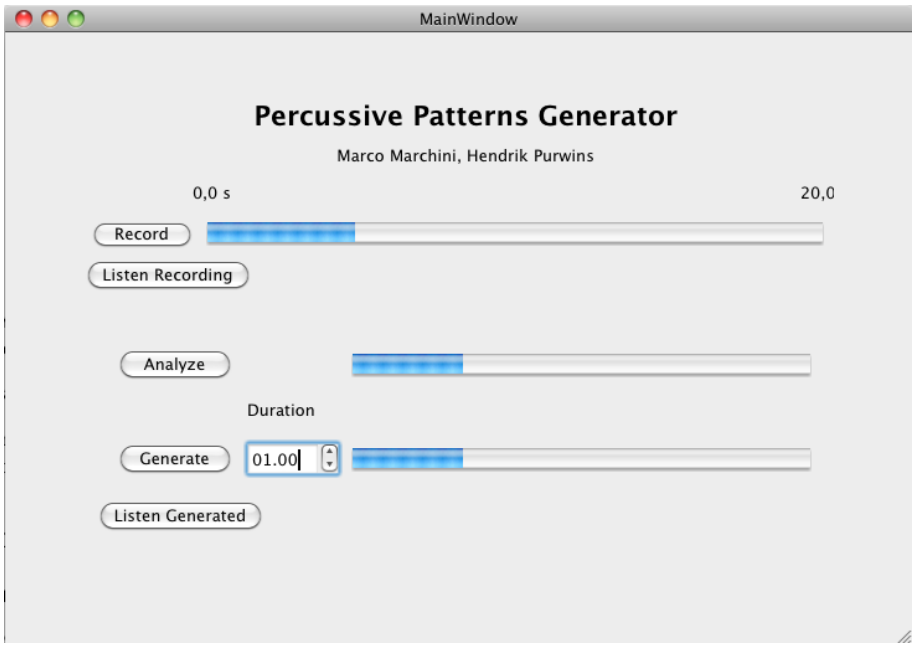

Figure B.1: Interface implemented in Qt4 where is possible to record percussive audio file, analyze them and generate continuations of the desired duration.

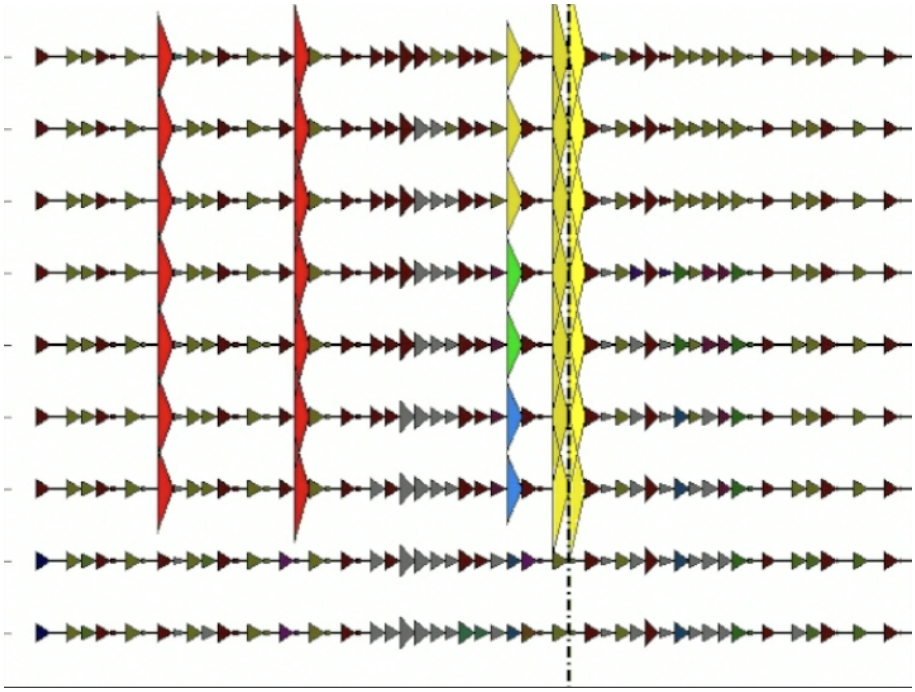

Figure B.2: Frame capture from a video representing the generated output of the system.

Bibliography

[1] www.youtube.com/user/audiocontinuation, April 2010.

[2] Samer Abdallah and Mark Plumbley. Information dynamics: patterns of expectation and surprise in the perception of music. *Connect. Sci*, 21(2-3):89–117, 2009.

[3] A.R. Addessi, L. Ferrari, S. Carlotti, and F. Pachet. Young children's musical experiences with a flow machine. In *Proceedings of the 9th International Conference on music perception and cognition*, 2006.

[4] G. Assayag, G. Bloch, M. Chemillier, A. Cont, and S. Dubnov. Omax brothers: a dynamic yopology of agents for improvization learning. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, page 132. ACM, 2006.

[5] Peter Buhlmann and Abraham J. Wyner. Variable length markov chains. *Annals of Statistics*, 27:480–513, 1999.

[6] G. Castellano, M. Mortillaro, A. Camurri, G. Volpe, and K. Scherer. Automated analysis of body movement in emotionally expressive piano performances. *Music Perception*, 26(2):103–119, 2008.

[7] R. Cilibrasi, P. Vitányi, and R. Wolf. Algorithmic clustering of music based on string compression. *Computer Music Journal*, 28(4):49–67, 2004.

[8] David Cope. *Virtual Music: Computer Synthesis of Musical Style*. MIT Press, Cambridge, Massachusetts, 2004.

[9] S. Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39–58, 2001.

[10] S. Dubnov, G. Assayag, and A. Cont. Audio oracle: A new algorithm for fast learning of audio structures. In *Proceedings of International Computer Music Conference (ICMC)*, pages 224–228, 2007.

[11] S. Dubnov, G. Assayag, and R. El-Yaniv. Universal classification applied to musical sequences. In *Proceedings of the International Computer Music Conference*, pages 332–340, 1998.

[12] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern classification*. Citeseer, 2001.

[13] Olivier Gillet and Gaël Richard. Enst-drums: an extensive audio-visual database for drum signals processing. In *ISMIR*, pages 156–159, 2006.

[14] A. Hazan, R. Marxer, P. Brossier, H. Purwins, P. Herrera, and X. Serra. What/when causal expectation modelling applied to audio signals. *Connection Science*, 21:119 – 143, 2009.

[15] IRCAM. Fishing game: www.youtube.com/watch?v=uswsw5ulssc.

[16] O. Lartillot, P. Toiviainen, and T. Eerola. A matlab toolbox for music information retrieval. In *Annual Conference of the German Classification Society*, 2007.

[17] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. The MIT Press, June 1996.

[18] Marco Marchini and Hendrik Purwins. Unsupervised generation of percussion sound sequences from a sound example. In *Sound and Music Computing Conference*, 2010.

[19] Ricard Marxer and Hendrik Purwins. Unsupervised incremental learning and prediction of audio signals. In *Proceedings of 20th International Symposium on Music Acoustics*, 2010.

[20] L.B. Meyer. *Style and music: Theory, history, and ideology*. University of Chicago Press, 1996.

[21] F. Pachet. Playing with virtual musicians: the continuator in practice. *IEEE Multimedia*, 2002.

[22] Francois Pachet. The continuator: Musical interaction with style. In ICMA, editor, *Proceedings of ICMC*, pages 211–218. ICMA, September 2002.

[23] H. Purwins, P. Holonowicz, and P. Herrera. Polynomial extrapolation for prediction of surprise based on loudness - a preliminary study. In *Sound and Music Computing Conference*, Porto, 2009.

[24] Hendrik Purwins, Maarten Grachten, Perfecto Herrera, Amaury Hazan, Ricard Marxer, and Xavier Serra. Computational models of music perception and cognition II: Domain-specific music processing. *Physics of Life Reviews*, 5:169–182, 2008.

[25] Dana Ron, Yoram Singer, and Naftali Tishby. The power of amnesia: learning probabilistic automata with variable memory length. *Mach. Learn.*, 25(2-3):117–149, 1996.

[26] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, January 2001.

[27] B.W. Vines, C.L. Krumhansl, M.M. Wanderley, and D.J. Levitin. Cross-modal interactions in the perception of musical performance. *Cognition*, 101(1):80–113, 2006.

[28] B.W. Vines, M.M. Wanderley, C.L. Krumhansl, R.L. Nuzzo, and D.J. Levitin. Performance gestures of musicians: What structural and emotional information do they convey? *Gesture-based communication in human-computer interaction*, pages 3887–3887, 2003.

[29] Marcelo J. Weinberger, Jorma J. Rissanen, and Meir Feder. A universal finite memory source. *IEEE Trans. Inf. Theory*, 41(3):643–652, 1995.

Acknowledge

Many thanks to Panos Papiotis for his patience during lengthy recording sessions and for providing us with beat boxing examples, the evaluation feedback, and inspiring comments. Many thanks also to Vassilis Pantazis for participating to the evaluation questionnaire. Big thanks to Ricard Marxer for his helpful support. All the my gratitude goes to Mirko Degli Esposti and Anna Rita Addessi for their support and for motivating this work.