Unsupervised structure analysis of an audio file

using Hierarchical Dirichlet Process Hidden Markov Models

Project Report

Jose Luis Diez Antich
Mattia Paterna

Aalborg University Copenhagen
Sound and Music Computing

This document has been designed in LaTeX. All the scripts, simulations and plot have been done using Mathworks' MATLAB software. Every effort has been made to ensure that all the information presented is correct.

Sound and Music Computing
Aalborg University Copenhagen
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Unsupervised structure analysis of an audio file using Hierarchical Dirichlet Process Hidden Markov Model

Theme:
Music information retrieval

Project Period:
Spring Semester 2016

Project Group:
IX

Participant(s):
Jose Luis Diez Antich
Mattia Paterna

Supervisor(s):
Hendrik Purwins

Copies: 1

Page Numbers: 47

Date of Completion:
July 28, 2016

Abstract:

A system for unsupervised learning of musical structures using Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM) is presented. It has been used in conjunction with previous approaches, whose flow is as follow: 1) segmentation by onset detection, 2) timbre representation by Mel-Frequency Cepstrum Coefficients, 3) symbolization using *single-linkage* clustering, 4) segments homogenization using the process by Marchini and Purwins. In addition, our system presents: 5) MFCC extraction over the homogenized segments and 6) unsupervised learning using HDP-HMM. Such built system has been evaluated using two percussive audio file from the ENST database. The evaluation part has used the Adjusted Random Index (ARI) to compare the HDP-HMM output state sequence with the result of the homogenified original structure.

Lyd og Musik
Aalborg Universitet København
http://www.aau.dk

AALBORG UNIVERSITET
STUDENTERRAPPORT

Titel:
Unsupervised structure analysis of an audio file using Hierarchical Dirichlet Process Hidden Markov Model

Tema:
Music information retrieval

Projektperiode:
Forårssemestret 2016

Projektgruppe:
IX

Deltager(e):
Jose Luis Diez Antich
Mattia Paterna

Vejleder(e):
Hendrik Purwins

Oplagstal: 1

Sidetal: 47

Afleveringsdato:
28. juli 2016

Abstract:

A system for unsupervised learning of musical structures using Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM) is presented. It has been used in conjunction with previous approaches, whose flow is as follow: 1) segmentation by onset detection, 2) timbre representation by Mel-Frequency Cepstrum Coefficients, 3) symbolization using *single-linkage* clustering, 4) segments homogenization using the process by Marchini and Purwins. In addition, our system presents: 5) MFCC extraction over the homogenized segments and 6) unsupervised learning using HDP-HMM. Such built system has been evaluated using two percussive audio file from the ENST database. The evaluation part has used the Adjusted Random Index (ARI) to compare the HDP-HMM output state sequence with the result of the homogenified original structure.

Contents

Preface

This report has been prepared in partial fulfilment of the requirement for the theme: *Music Information Research* in Sound and Music Computing, 2nd semester in the Academic year 2015-2016

The report has been written in May 2016 in a four-month semester project under the supervision of Dr. Hendrik Purwins.

We would like to thank Marco Marchini for assisting us in understanding his code. To Ricard Marxer for the his very useful suggestions in understanding the basis of Hierarchical Dirichlet Process Hidden Markov Model. To Emily Fox for answering our doubts. Finally, we would like to give special thanks to our supervisor, Hendrik Purwins, for his help and dedication throughout the developing of the project.

Aalborg University, July 28, 2016

Jose Luis Diez Antich
<jdieza15@student.aau.dk>

Mattia Paterna
<mpater15@student.aau.dk>

Chapter 1

Introduction

Recent years have seen enormous advances in the use of information technology related to *music information retrieval* (MIR) [23]. Great efforts have been directed toward the development of techniques for searching and extracting useful information from, for instance, *waveform-based* music data. This information can be then used, for example, to identify different instruments, or provide a continuation to an audio sequence, etc.

The mentioned applications can be achieved using either supervised or unsupervised machine learning methods. The first method requires previous knowledge of the data (the *training set*) in order to classify new data (the *testing set*), as opposite of the unsupervised methods, which do not require previous knowledge. In the same way, these methods can be either learn the whole data at once (*batch learning*) or use each data point in a sequentially (*incremental learning*)[21].

The motivation for this project is to learn more about these machine learning methods with the purpose of, as a future work, developing a system which a performer could improvise with.

This project can be understood as the first step towards understanding a recent and promising method of unsupervised learning, the Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM). This first step consists in understanding how its clustering process works, in particular how this method analyzes the structure of a musical piece. The next step could analyze how this method is used to generate new data, in particular how to create a continuation of a musical piece. These two first steps approach the learning with a batch learning manner, whereas a further third step could be to apply the previous steps with an incremental learning approach.

To sum up, this report is dealing only with the offline unsupervised structure analysis. The successive step, an offline audio continuation, is effectively described in [20], as well as [21], which achieves an online incremental clustering method.

1.1 Audio continuation

A definition The purpose of audio continuation is to generate a meaningful audio sequence based on an incoming audio signal. It should be *interesting*, first of all. However, as explained by Pachet, two distinct and incompatible classes could be defined in musical performances: *interactive* and *imitation* systems. The former were not able to learn but they can quickly transform musical inputs. On the other hand, the latter didn't support musical interaction and they needed human input for supervised learning.

Composition and music improvisation are both fascinating activities, whose the basic building blocks are recombinations of pitches and duration. One of the first formal types of algorithms in music history is the eighteenth-century *Musikalisches Würfelspiel* .The idea behind was to compose some music measures that could be recombined in several arbitrarily and randomly ways. The figured bass, a popular Baroque technique for writing music, uses combination of notated music, period style constraints and performer choice. Many more examples can be found in [3]. That means composition and improvisation has always been related and, with the advent of more sophisticated machines, the aim to create interesting musical material and to build computational architectures of musical sequence learning has become more ambitious.

Related work Pachet's work, named the *Continuator*, aimed to bridge the gap between interaction and imitation building operational representations of musical styles in a real time context [25]. The heart of his MIDI-based system relies on a Markov-based model of musical styles, its very spectacular application can be found in the automatic compositions of David Cope, which remind the pioneering *Illiac Suite* by Hiller and Isaacson [15] and the composition of Xenakis, which mostly have been based on algorithms, stochastic processes and computer-generated material that best fit his musical needs. Hazan et al. [14] built a causal system to represent a stream of music into musical events, and to generate further expected events. Their system is based on low-level (i.e. MFCC) and mid-level features extraction such as onsets and beats, and on an unsupervised clustering process that builds and maintains a set of symbols aimed at representing musical stream events using both timbre and time descriptions. They then use *Predictive Partial Match* to process those symbols. A similar system is also used in the works by Marchini and Purwins [20] and Marxer and Purwins [21]. While in the former the analysis is performed using *single-linkage* clustering and the final synthesis is performed employing Variable-length Markov Chain, recombining the audio material derived from the sample itself, in the latter extraction of statistical regularities of the symbol sequence, using hierarchical N-grams and the newly introduced conceptual *Boltzmann machine*. They all are based on Markov-chain models. However,

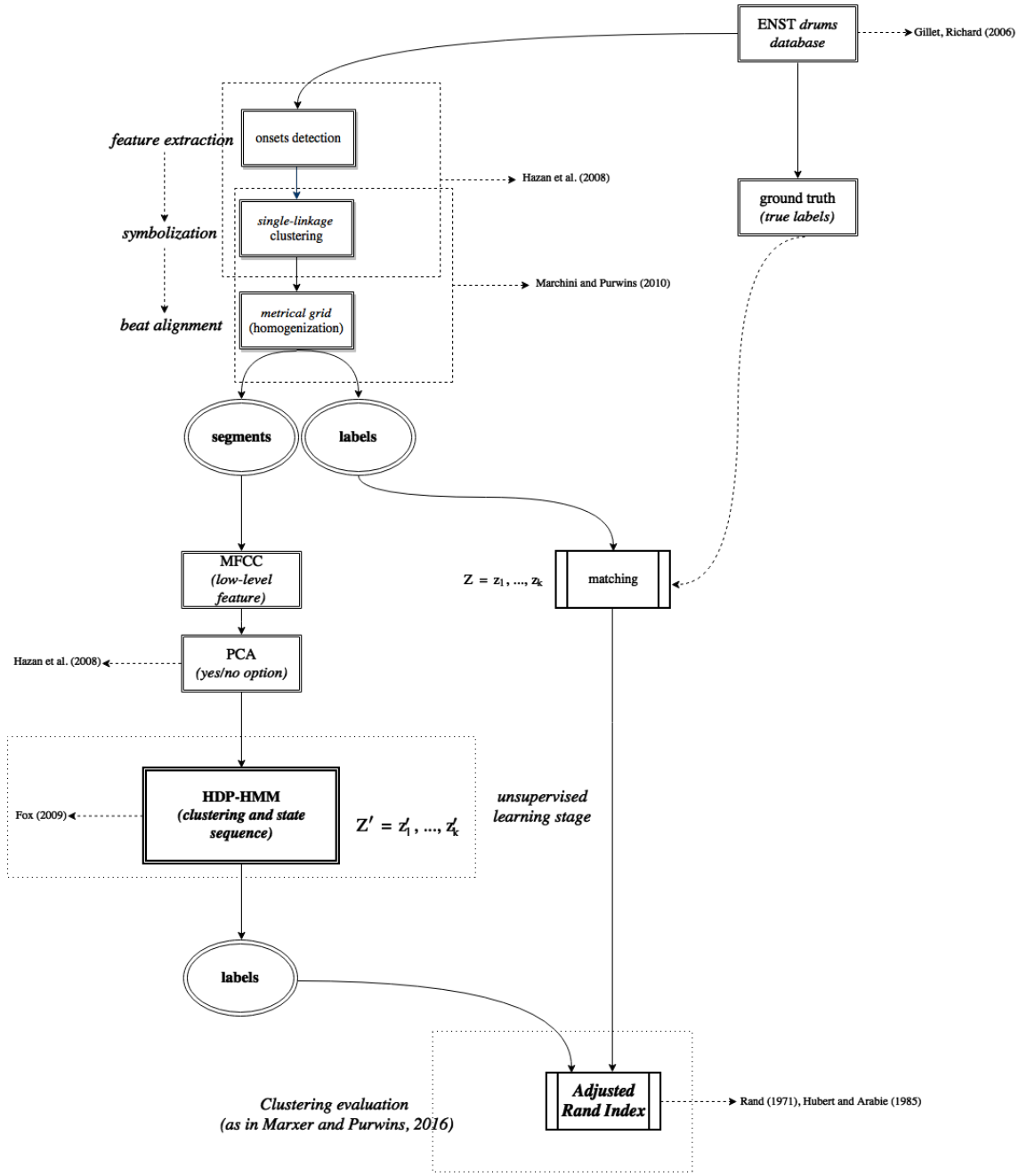


Figure 1.1: The overall framework of our project. It can be noticed in such sense it could be considered as the natural extension of the works of Hazan et al. [14], Marchini and Purwins [20].

the last work uses Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM) in conjunction with clustering for the segmentation process.

1.2 General description

Our analysis system, which is shown in 1.1 is based on the previous work by Marchini and Purwins [20] which will be described in detail in chapter 5. Such approach is *offline*, but several concepts are provided that are useful for a pre-processing stage and for the definition of a regular metrical structure, that is the *homogenization* section of that system. In doing so, much attention to the *music segmentation* process - that is cutting a whole sound file into smaller sections, each with some homogeneous properties with regard to the musical content - and *music structure analysis* - e.g. extracting possible patterns and structure repetitions, which yields to define a *musical* structure - has been devoted. Moreover, for the feature extraction part, the MATLAB MIR toolbox [18] has been used, since it provides an extended and powerful framework for processing, classification and retrieval.

As said before, the Hierarchical Dirichlet Process Hidden Markov Model is the statistical method used to analyze the music signal. For this reason, first, Hidden Markov Models are explained in chapter 2. Secondly, a theoretical investigation about Dirichlet processes (DP) and their applications has been conducted throughout chapter 3 and 4. The use of hierarchical DPs combined to HMM, named *hierarchical Dirichlet Process Hidden Markov Model* (HDP-HMM), is presented first in [30] and has been used by Fox et al. to address the problem of *speaker diarization* in [8]. Shortly, the HDP-HMM leads to data–driven learning algorithms which infer posterior distributions over the number of states [9], allowing for the number of states not to be set a priori.

In evaluating the whole system, some percussive sounds have been used. All the sound files come from the *ENST* database, that provided a collection of around forty drum recording examples (for more information, see [13]). Using percussive sounds allows us to avoid an excessive complexity of the waveform representation since it dramatically increases when considering polyphonic orchestral music, where the components of various musical tones interfere with each other. An evaluation on two musical audio files is given in chapter 6, as well as a short description of both the files. Finally, some conclusions over different values of the hyperparameters for the HDP-HMM process are drawn.

Chapter 2

Hidden Markov Models

Hidden Markov Models, or HMMs, are one of the most popular statistical technique that has been successfully applied to many fields with the purpose of model temporal patterns. These fields include speech recognition, speech synthesis, completing genomes, image classification, poem generation, etc.

HMMs are a particular case of Markov models, which are stochastic models used to model randomly changing systems where it is assumed that future states depend only on the current state and independent from previous events. The simpler case of Markov model is the Markov Chain.

In this chapter, an overview of Hidden Markov models is covered. It closely follows [26], [5], [24] and [22]. First, Markov Chains are described to introduce the description of the Hidden Markov Models.

2.1 Markov Chain

Three elements define a Markov Chain: a set of N states $Z = \{z_1, z_2, ..., z_n\}$, the transition probabilities between those states, $A = (a_{ij})_{i \in Z, i \in Z}$, and the probability of starting in each state, $\pi^0 : Z \to \{0, 1\} = \{\pi_1^0, \pi_2^0, ..., \pi_n^0\}$, [26].

Over a number of discrete time steps, $T = 1, 2, ...t$, a Markov Chain will output a sequence of states, in which the current state is denoted by q_t. The probability of switching to the next state, q_{t+1}, only depends on q_t, it is independent of the preceding states. This probability of having state z_j after having had state z_i can be expressed in the following way:

$$a_{ij} = P(q_{t+1} = z_j | q_t = z_i), 1 \leq i, j \leq N \tag{2.1}$$

In the seminal work by Rabiner [26], Markov Chains are illustrated with a system of three states which describe the weather of the day, $Z = \{'rain', 'cloudy', 'sunny'\}$.

Their transition probability matrix is the following:

$$A = \{a_{ij}\} = \begin{pmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{pmatrix} \tag{2.2}$$

One can see the probability of having a rainy day after another rainy day is 0.4. In the same way, it is 0.2 probable to have a sunny day after a cloudy day.

The initial probability set of the model is $\pi^0 = \{0, 0, 1\}$. That means, the only possible weather in the first day is $'sunny'$.

In this case, an interesting questions is: What is the probability of having an exact sequence of states? For example, consider the observation, or sequence of states, $Z = {}'sunny',' rain',' rain',' cloud'$. Its probability can be computed as:

$$\begin{aligned} P(O|Model) = P('sunny',' rain',' rain',' cloud'|Model) = \\ P(z_3, z_1, z_1, z_2|Model) = \\ P(z_3) \cdot P(z_1|z_3) \cdot P(z_1|z_1) \cdot P(z_2|z_1) = \\ \pi_3 \cdot a_{31} \cdot a_{11} \cdot a_{12} = \\ 1 \cdot 0.1 \cdot 0.4 \cdot 0.3 = 0.012 \end{aligned} \tag{2.3}$$

As said before, in the case of Markov Chains, the output of the system is the sequence of states, in other words, the observation of each state is the state itself.

2.1.1 Hidden Markov Models

In the case of Hidden Markov Models, the states are now a hidden stochastic process and, therefore, the output of a HMM systems is not a sequence of states, but a sequence of observations emitted from these states.

Markov chains are well suited in applications involving symbolic data. However, in the case where the data to model are feature vectors sequences describing audio, it is required a level of complexity that HMMs can offer [16]. HMMs assume that there is a set of states that generates the data. For example, in the case of this project, each state represents a musical instrument that generates a set of observations, in particular MFCC vectors. Each state is associated to an emission probability density function that generates the observations.

Formally, HMMs define a probability distribution over sequences of observations $Y = \{y_1, y_2, ..., y_t\}$, emitted from a set of K states $Z = \{z_1, z_2, ..., z_K\}$. The model is defined in terms of the *state-specific transition distribution*, π_i, of the *emission parameters*, θ_i for state i and of the initial state distribution π^0.

With this notation, the probability of switching to state z_{t+1} after having had state z_t is distributed according to π_{z_t}:

$$z_{t+1}|z_t \sim \pi_{z_t} \tag{2.4}$$

In the same way, the observations are generated as a function of the emission parameters of the state, θ:

$$y_t|z_t \sim F(\theta_{z_t}) \tag{2.5}$$

Where θ_{z_t} represent the emission parameters for state z_t.
Figure 2.1 shows a graphical representation of a HMM over n time steps.

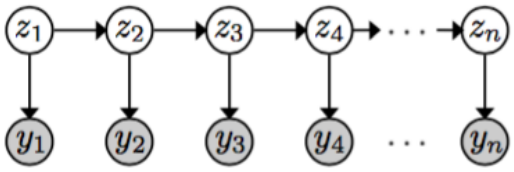


Figure 2.1: HMM diagram taken from [7]. Top row shows a sequence z_i of n states and the bottom row shows the sequence of y_i observations drawn from these states.

2.1.2 Basic problems for Hidden Markov Models

The way that HMMs are used in real world applications can be summarized with three basic problems. In this section, these problems are superficially defined, for more details, please use HMMs seminal work [26]. These problems are:

1. Given a HMM model, what is the probability of observing a particular sequence of symbols?
   This problem is useful in the case of, given a set of HMM models, finding the model that most likely has generated an observation sequence. To solve it, the Maximal Likelihood algorithm is used.

2. Given the observations, find the most probable state sequence.
   This problem is the most related to this project, its goal is to unveil the hidden part of the model. In other words, finding the state sequence that most likely has produced the observations.
   In this project, and as said before, the observations are the MFCC vectors. Therefore, the solution for this problem, would be the sequence of instruments that has produced the MFCC vectors. To find the best sequence of states, it is popular to use the Viterbi algorithm.

3. Adjust model parameters to maximize an observations sequence.
   This problem involves using the given observation sequence to make the model learn, or optimizing the model parameters to fit the sequence. This problem is useful to create models to real data.
   Since there is no known analytical way to solve this problem, procedures such as the Baum-Welch method are used to find the local maxima of the parameters.

2.1.3 Representation of HMM in Continuous Space

So far, HMM have been described in the finite integer space, i.e. the number of states of the HMM is finite and has to be predefined. As said in the introduction, the goal of the unsupervised system of this project is to learn from the data how many states have generated the data. In order to achieve this, it is necessary to move from the integer space to the parameter space Θ.

The equivalent representation of HMMs in the parameter space Θ is via a set of *transition probability measures*:

$$G_j = \sum_{k=1}^{K} \pi_{jk} \delta_{\theta_k} \tag{2.6}$$

where δ_θ is a unit mass centered at θ, the emission parameters and π is the weight associated to it.

In the previous representation, each state contained a different set of emission parameters, it can be understood that the state was used as an index to obtain these parameters. With this representation, there is no need to use these *indices*, because the parameter space Θ is used directly. And, therefore, the draws from G_j are the emission parameters, $\theta' \in \{\theta_1, ..., \theta_K\}$, as the following equations formalize:

$$\theta'_t | \theta'_{t-1} \sim G_{j_{t-1}} \tag{2.7}$$

$$y_t | \theta'_t \sim F(\theta'_t) \tag{2.8}$$

The equations above are the equivalent to equations 2.4 and 2.5 in the previous notation.

Once, the HMM system is represented in the parameter space, to consider a *Bayesian HMM*, the transition probability measures G_j have to be treated as random measures and provide them with a prior distribution [5]. First, the weights $\pi_j = [\pi_{j1}, ...\pi_{jK}]$ are taken as independent draws from a K-dimensional Dirichlet distribution with concentration parameters $\alpha_1, ..., \alpha_K$:

$$\pi_j \sim Dir(\alpha_1, ..., \alpha_K) j = 1, ..., K. \tag{2.9}$$

And, second, the emission parameters, θ are taken as draws from a base measure H, which can be a Gaussian distribution, on the parameter space Θ:

$$\theta_j \sim H \tag{2.10}$$

Even tough HMM have been a successful tool for a large number of applications, one significant limitation is related to the number of states in the system. The number of states is fixed and has to be predefined [5]. In the next chapters, *Dirichlet Process* and *Hierachical Dirichlet Process* are described, which lead to the extension of the Bayesian HMM to *Hierachical Dirichlet Process - Hidden Markov Model*, in which the limitation of the fixed states is no longer.

Chapter 3

Dirichlet Processes

Probabilistic models are used throughout machine learning to model distributions over observed data. Traditional parametric models using a fixed and finite number of parameters can suffer from over- or under-fitting of data.

A process is a collection of random variables indexed by some set, where all the random variables are defined over the same underlying set and the collection of random variables has a joint distribution[11].

The Dirichlet process is a stochastic process used in Bayesian nonparametric models of data. It is a distribution over distributions, that is each draw from a Dirichlet process is itself a distribution. The Dirichlet process is currently one of the most popular Bayesian non- parametric models [28]. Finally, the Dirichlet process has been useful in developing flexible models for a wide variety of problems. Although the discrete nature of the DP makes it unsuitable for general applications in Bayesian nonparametrics, it is well suited for the problem of placing priors on mixture components in mixture modeling [30].

Some different characherization of the Dirichlet process will be presented throughout this chapter, as well as its main definition and some applications. This chapter closely follows the book by Gelman et al. [12], the works of Fox, Sudderth, Jordan and Willsky [10] [9] [5], the work by Teh [28] and the work by Frigyik et al. [11].

3.1 Definition

The Dirichlet process (DP) is a stochastic process whose sample paths are probability measures with probability one. [28] In other words, the Dirichlet process, denoted by DP(γ, H), provides a distribution over discrete probability measures with an infinite collection of atoms [5]

$$G = \sum_{\kappa=1}^{\infty} \beta_\kappa \delta_{\theta_\kappa} \qquad \theta_k \sim H \tag{3.1}$$

on a parameter space Θ [10]. In this equation, δ_θ is the point mass centered at θ and β is the mixing proportion.

Stochastic processes are distributions over function spaces, with sample paths being random functions drawn from the distribution. In the case of the DP, it is a distribution over probability measures.

To explain further, for a random distribution G to be distributed according to a DP its marginal distributions have to be Dirichlet distributed. Specifically, let H be a distribution over θ and γ be a positive real number. Then for any finite measurable partition $A_1, ..., A_k$ of Θ the vector $(G(A_1), ..., G(A_k)$ is random since G is random. Hence, G is Dirichlet process distributed with base distribution H and concentration parameter γ

$$G \sim DP(\gamma, H) \tag{3.2}$$

if

$$(G(A_1), ..., G(B_k)) \sim Dir(\gamma H(A_1), ..., \gamma H(A_k)) \tag{3.3}$$

for every finite measurable partition $A_1, ..., A_k$ of Θ.

3.2 Prior and posterior Dirichlet process

Let a sample space Θ partitioned into measurable subsets $A = \{A_1, ..., A_k\}$. If the sample space $\in \Re$, then $A_1, ..., A_k$ are simply non-overlapping intervals partitioning the real line into a finite number of bins.[1]

Let G denote the unknown probability measure over (Θ, B), with B the collection of all possible subsets of the sample space Θ. If G is a random probability measure (RPM), then these bin probabilities are random variables. A simple conjugate prior for the bin probabilities corresponds to the Dirichlet distribution [12]:

$$G(A_1), ..., G(A_k) \sim Dir(\gamma H(A_1), ..., \gamma H(A_k)) \tag{3.4}$$

where H is a base probability measure providing an initial guess at G, and γ is a prior concentration parameter controlling the degree of shrinkage of G toward H [12].

Unfortunately, that equation specifies that bin A_k is assigned probability $G(A_k)$ and does not specify how probability mass is distributed across each bin. The idea is to induce a fully specified prior for G for all possible partition B avoiding sensitivity in the choice of partition [12]. For this specification to be coherent, there

[1] This section is entirely based on chapter 23.2 in Gelman et al. [12] and on the work of Teh [28]

must exist a random probability measure G such that the probability assigned to any measurable partition $A_1, ..., A_k$ by G is $Dir(\gamma H(A_1), ..., \gamma H(A_k))$. That resulting random probability measure G is referred to as a Dirichlet process. This H baseline probability measure is commonly chosen to correspond to a parametric model such as a Gaussian [12].

The definition of the Dirichlet process and properties of the Dirichlet distribution imply:

$$G(A) \sim Beta(\gamma H(A), \gamma(1 - H(A))) \tag{3.5}$$

for A_k belonging to the collection of all possible subsets of B.

(For further information about Dirichlet distribution and its properties see [11])

The parameters H and γ play intuitive roles in the definition of the Dirichlet process [28]. That is, the base distribution H is basically the mean of the DP and the concetration parameter can be understood as the inverse variance [28]. It follows that the prior mean has the form

$$E[G(A)] = H(A) \tag{3.6}$$

so that the prior for G is centered on H [12]. In addition, the prior variance is

$$var[G(A)] = \frac{H(A)(1 - H(A))}{\gamma + 1} \tag{3.7}$$

The larger γ is, the smaller the variance, and the DP will concentrate more of its mass around the mean [28].

The DP prior distribution also has a conjugacy property which makes inferences straightforward:

$$G|z_1, ...z_n \sim DP\left(\gamma H + \sum_i \delta_{x_i}^z\right) \tag{3.8}$$

where $z_i \sim G$ and draws are independent and identically distributed (IID) [12].

The updated precision parameter is $\gamma + n$, so that γ is in some sense a prior sample size. The posterior expectation of G is defined as

$$E[G(A)|z^n] = \left(\frac{\gamma}{\gamma + n}\right) H(A) + \left(\frac{n}{\gamma + n}\right) \sum_{i=1}^{n} \frac{1}{n} \delta_{z_i} \tag{3.9}$$

In the limit as the precision parameter γ approaches 0, in some sense the prior distribution is just non-informative. On the other hand, as the amount of observation grows so that $n \gg \gamma$ the posterior is just dominated by the empirical distribution, which is a *close* approximation of the true underlying distribution [28].

It can be noticed how the posterior base distribution is a weighted average between the prior base distribution and the empirical distribution. The weight

associate with the prior distribution is strictly proportional to γ while the empirical distribution is proportional to the number of observation n. Thus we can interpret γ as the mass associated with the prior, that is its *strength* [28].

This finally yields to a relevant property of the Dirichlet process: the posterior DP approaches the true underlying distribution as the number of observation increases.

3.3 Generating samples from a Dirichlet process

In this section the stick-breaking, the Pólya urn process and the Chinese restaurant process are explained, of which the latter two are different names for the same process. In fact, they both refer to the same process due to Blackwell-McQueen. In all cases, samples are draws generating first the sequences $\{\beta_k\}$ and $\{\theta_k\}$, then using eq. 3.1 to produce a sample measure G. The trouble is drawing the $\{\beta_k\}$ partly because they are not independent of each other, since they must sum to one.

3.3.1 Predictive distribution: the Pólya urn model

Let us consider drawing a $G \sim DP \sim (\gamma, H)$ and drawing some observations $\theta_1, ..., \theta_n$ from the previous distribution. It can be demonstrated that the posterior base distribution given $\theta_1, ..., \theta_n$ is also the predictive distribution of $\theta_{n+1}|G, \theta_1, ..., \theta_n \sim G$ [28]:

$$\theta_{n+1}|G, \theta_1, ..., \theta_n = \frac{1}{\gamma + n}\left(\gamma H + \sum_{i=1}^{n} \delta_{\theta_i}\right) \tag{3.10}$$

The sequence of predictive distribution $\theta_1, ..., \theta_n$ is called *Blackwell-McQueen urn scheme*. They derived a Pólya urn representation, where the metaphor is really useful in interpreting. Specifically, each value in Θ is a unique color, and draws $\theta \sim G$ are balls with the drawn value being the color of the ball. Aside there is also a urn containing previously seen balls. In the beginning there are no balls in the urn. A color drawn from H is picked, a ball is painted with that color and then dropped in the urn. At step $n + 1$ the probability to pick a new color will be $\frac{\gamma}{\gamma+n}$ while the probability to pick a color that the urn already contains will be $\frac{n}{\gamma+n}$ [28].

A relevant property of the predictive distribution in 3.10 is that it has point masses located at the previous draws $\theta_1, ..., \theta_n$. Moreover, for a long enough sequence of draws from G, the value of any draw will be repeated by another draw, implying that G is composed only of a weighted sum of point masses, that said G is a discrete distribution [28].

3.3.2 Stick-breaking construction

Since draws from a DP are composed of a weighted sum of point masses, it is possible to provide a direct constructive definition of the DP as such. This construction is a more straightforward proof of the existence of the Dirichlet processes and is named *stick-breaking (Sethuraman) construction*. This also is useful in obtaining further insight into properties of the DP and as a stepping stone for generalizations.

Recalling the definition of the DP in 3.1, the weights are sampled using this construction [10] as shown in the equation:

$$\beta_k = v_k \prod_{n=1}^{k-1} (1 - v_n) \tag{3.11}$$

where $v_k \sim \ Beta(1, \gamma)$.

This distribution can be also denoted by $\beta \sim GEM(\gamma)$.

It can be noted how this construction guarantees that the weights sum to 1. In fact, the stick-breaking process starts with a stick of unit length representing the total probability to be allocated to all the atoms. Initially, a random piece of length v_1 is selected, with the length generated from a $Beta(1, \gamma)$ distribution, and this probability weight v_1 is allocated to the randomly generated first atom $\theta_1 \sim \ G$. There is then $1 \ \ v_1$ of the stick remaining to be allocated to the other atoms. The more the process goes on, the shorter the sticks get so that the lengths allocated to the higher indexed atoms decrease stochastically, with the rate of decrease depending on the DP precision parameter γ [12].

Drawbacks Stick-breaking draws the β_k exactly, but one at a time: β_1, then β_2, etc. Since there are an infinite number of β_K, it takes an infinitely long time to generate a sample. If stick-breaking is stopped early, then only the first $k1$ coefficients are exactly correct [11].

3.3.3 The Chinese Restaurant Process

A Chinese restaurant process metaphor is commonly used in describing the Polya urn scheme [12]. That comes from the clustering property which 3.10 implies [28]. The discreteness and clustering properties of the DP play crucial roles in the use of DPs for clustering via DP mixture models, one of its possible application.

Since the values of the draws can be repeated due to the smoothness, let $\theta_1^*, ..., \theta_m^*$ be unique values among $\theta_1, ..., \theta_n$ and n_k be the number of repeats of θ_k^*. The predictive distribution can be equivalently rewritten as:

$$\theta_{n+1} | G, \theta_1, ..., \theta_n \sim \frac{1}{\gamma + n} \left(\gamma H + \sum_{i=1}^{m} n_k \delta_{\theta_k^*} \right) \tag{3.12}$$

	indexing	Partition Labels
Pólya Urn	sequence of draws of balls	ball colors
Chinese Restaurant	sequence of incoming customers	different dishes
Clustering	sequence of natural numbers	clusters

Table 3.1: Equivalences between different descriptions of the same process. Table taken from [11]

In this equation, the value θ_k^* will be repeated by θ_{n+1} with probability proportional to n_k, the number of times it has already been observed. The larger n_K is, the higher the probability that it will grow. This is a rich-gets-richer phenomenon, where large clusters (a set of θ_i's with identical values θ_k^* being considered a cluster) grow larger faster [28].

That said, the distribution on partitions induced by the sequence of conditional distributions in 3.12 is commonly referred to as the *Chinese restaurant process* [10]. Consider a restaurant with infinitely many tables. The first customer sits at a table with dish θ_i^*. The second customer sits at the first table with probability $\frac{\gamma}{1+\gamma}$ or a new table with probability $\frac{1}{1+\gamma}$. This process continues with the ith customer sitting at an occupied table with probability proportional to the number of previous customers at that table and sitting at a new table with probability proportional to γ. Each occupied table in the (infinite) restaurant represents a different cluster of subjects, with new clusters added at a rate proportional to $\gamma \log n$. The number of clusters depends (probabilistically) on the number of subjects n with new clusters introduced as needed as additional subjects are added to the sample [12].

From the Chinese restaurant process, it is noticeable that the Dirichlet process has a reinforcement property that leads to a clustering at the values θ_k. This representation also provides a means of sampling observations from a Dirichlet process without explicitly constructing the infinite probability measure $G \sim DP(\gamma, H)$ [10]. Finally, as expressed in [28] γ controls the number of clusters in a direct manner, with larger γ implying a larger number of clusters *a priori*. This intuition will help in the application of DPs to mixture models.

3.4 Applications

Because of its simplicity, DPs are used across a wide variety of applications of Bayesian analysis in both statistics and machine learning. The DP is commonly used as a prior on the parameters of a mixture model of unknown complexity [9]. Therefore, one of the most prevalent applications include clustering *via mixture models*. Here, the nonparametric nature of the Dirichlet process translates to mixture models with a countably infinite number of components. These models provide an alternative to methods that attempt to select a particular number of mixture components, or methods that place an explicit parametric prior on the number of

components. [30]

3.4.1 Dirichlet process mixture models

A set of observation $\{z_1, ..., z_n\}$ could be modeled using a set of latent parameters $\{\theta_1, ..., \theta_n\}$. Each θ_i is drawn independently and identically from G, while each z_i has distribution $F(\theta_i)$ parametrized by θ_i:

$$\begin{aligned} z_i|\theta_i &\sim F(\theta_i) \\ \theta_i|G &\sim G \\ G|\gamma, H &\sim DP(\gamma, H) \end{aligned} \tag{3.13}$$

Because G is discrete, multiple θ_i's can take on the same value simultaneously, and the above model can be seen as a mixture model, where z_i's with the same value of θ_i belong to the same cluster [28]. Moreover, θ_i^* is a specific value selected from G and associated with observation z_i. Particularly, $\theta \sim G$ is not the observed data, but rather a parameter (or a set of parameters) for some distribution $F(\theta)$, from which the observations are drawn. An example of this process is the Gaussian mixture model where $\theta = \{\mu, \}$, the mean and the covariance matrix for a multivariate normal distribution, and H the normal-Wishart conjugate prior. Whereas samples from the discrete G will repeat with high probability, samples from $F(\theta)$ are again from a continuous distribution. Therefore, values that share parameters are clustered together in that they exhibit similar statistical characteristics according to some distribution function, $F(\theta)$.

The mixture perspective can be even made more in agreement with the usual representation of mixture models using the stick-breaking construction. Let z_i be a cluster assignment variable, which takes on value k with probability π_k. Then eq. 3.14 can be equivalently expressed as:

$$\begin{aligned} \pi|\gamma &\sim GEM(\gamma) \\ \theta_i^*|H &\sim H \\ x_i|z_i, \{\theta_i^*\} &\sim F(\theta_{z_i}^*) \end{aligned} \tag{3.14}$$

and, recalling the stick-breaking process

$$G = \sum_{\kappa=1}^{\infty} \pi_\kappa \delta_{\theta_\kappa^*} \tag{3.15}$$

where π is the mixing proportion, θ_k^* are the cluster parameters, $F(\theta_k^*)$ is the distribution over data in cluster k and H the prior over cluster parameters.

The DP mixture model is an *infinite* mixture model, that is a mixture model with a countably infinite number of clusters and π_k decreasing in a exponential

manner yielding to a small number of cluster used to model the data a priori [28]. In the DP mixture model, the actual number of clusters used to model data is not fixed, and can be automatically inferred from data using a Bayesian posterior inference framework.

The Dirichlet process has also been used in more complex models involving more than one random probability measure. For instance, in multitask settings each task might be associated with a probability measure with dependence across tasks implemented using a hierarchical Bayesian model. In this last case, the most appropriate model to choose is a *hierarchical* Dirichlet process.

Chapter 4

Hierarchical Dirichlet Process Hidden Markov Model

As said in chapter 2, Hidden Markov Models (HMMs) require the number of hidden states in the system to be set a priori [8]. This limitation is avoided with the Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM), which is a Bayesian non parametric extension of the classical HMMs and allows to infer posterior probability over the cardinality of the hidden space.

The first section of this chapter describes Hierarchical Dirichlet Processes, and the second section describes the bayesian extension to HMM using them as a prior, i.e. Hierarchical Dirichlet Process Hidden Markov Model. For more details, the works of Teh and Jordan [30] [29], as well as the works of Fox [8] [10] [9], which this chapter follows closely, are suggested to the reader.

4.1 Hierarchical Dirichlet Processes

4.1.1 Definition

The hierarchical Bayesian extension of Dirichlet processes mixture models is naturally derived with the assumption that observations are produced by a related, yet distinct generative process [9].

A hierarchical Dirichlet process (HDP) defines a collection of probability measures $\{G_j\}$ on the points that represent the emission parameters $\{\theta_1, \theta_2, ...\}$ by assuming that each discrete measure $\{G_j\}$ is a variation on a global discrete measure G_0. [10]. To be more specific, the Bayesian hierarchical specification takes $G_j \sim DP(\alpha, G_0)$ with G_0 itself a draw from a Dirichlet process $DP(\gamma, H)$.Therefore, in this hierarchical model, G_0 is atomic and random. Letting G_0 be a base measure for the draw $G_j \sim DP(\alpha, G_0)$ implies that only these atoms can appear in G_j. Thus, atoms can be shared among the collection of the possible random measures $\{G_j\}$

[29].

These recursive construction have the effect of constraining the random measure G to place the atoms at the discrete locations determined by G_0 [29]. With this construction, the probability measures are therefore described as:

$$\begin{aligned} G_0 &= \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k} \qquad \beta|\gamma \sim GEM(\gamma) \\ G_j &= \sum_{k=1}^{\infty} \pi_{jk} \delta_{\theta_k} \quad \pi_j|\alpha, \beta \sim DP(\alpha, \beta) \end{aligned} \tag{4.1}$$

with

$$\theta_k|H \sim H \tag{4.2}$$

The random measures G_j are conditionally independent given G_0, with distribution given by a DP with base probability measure G_0:

$$G_j|\alpha, G_0 \sim DP(\alpha, G_0) \tag{4.3}$$

The basic notion of hierarchical DP is a specific example of a dependency model for multiple Dirichlet Processes that aims at the problem of sharing clusters among related groups of data.

As shown in fig. 4.1 the hyperparameters of the hierarchical DP consist of the baseline probability measure H and the concentration parameters γ and α. The baseline H provides the prior distribution for the factors θ'_{ji}. The distribution G_0 varies around the prior H with the amount of variability set by γ. The distribution G_j over the factors in the jth group deviates from G_0 depending on α [30].

Hierarchical DP mixture models A hierarchical DP can be used as the prior distribution over the factors for grouped data. For each j, let $\theta'_{j1}, \theta'_{j2}, ...$ be independently identically distributed random variables drawn from G_j. Each θ'_{ji} is a factor corresponding to a single observation y_{ji}. The likelihood is given by

$$\begin{aligned} \theta'_{ji}|G_j &\sim G_j \\ y_{ji}|\theta'_{ji} &\sim F(\theta'_{ji}) \end{aligned} \tag{4.4}$$

that is the complete definition of a *hierarchical DP mixture models*, whose graphical model is also shown in fig. 4.1.

The hierarchical DP can readily be extended to more than two levels. That is, the base measure H can itself be a draw from a DP, and the hierarchy can be extended for as many levels as are deemed useful. With this configuration, it is possible to

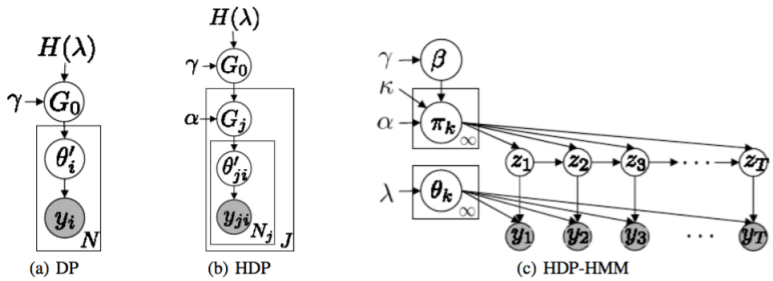


Figure 4.1: Dirichlet process (left),Hierarchical Dirichlet process (center) mixture models and HDP-HMM dynamic Bayesian network(right). Image taken from [4].

configure a tree in which a DP is associated with each node, in which the children of a given node are conditionally independent given their parent, and in which the draw from the DP at a given node serves as a base measure for its children [30].

4.1.2 The Chinese Restaurant Franchise

An analog of the Chinese restaurant process for hierarchical Dirichlet processes called the Chinese Restaurant Franchise (CRF) could be considered for a straightforward explanation of this process. It has been named in the seminal work of Teh, Jordan, Beal and Blei [30].

In this scenario, a restaurant franchise has a shared menu across its restaurants. At each table of each restaurant, one dish is ordered from the menu by the first customer who sits there, and this dish is shared among all of the customers who sit at that table. That said, multiple tables in multiple restaurants can serve the same dish. In this setup, the restaurants correspond to HDP groups and the customers correspond to the factors θ'_{ji}, while $\{\theta_1, ..., \theta_k\}$ denote a series of random variables distributed according to the baseline H: this is the global menu of dishes for the whole franchise, each of these dishes served in an *infinite* buffet line manner. Teh et al. also introduced new variables, ψ_{jt} that represent the table-specific choice of dishes, e.g. the dish served at table t in restaurant j. Finally, it can be seen how each θ'_{ji} is associated with one ψ_{jt}, whereas each ψ_{jt} is associated with one θ_k.

Each customer is preassigned to a given restaurant determined by the customer's group j. Upon entering the jth restaurant in the CRF, customer y_{ji} sits at currently occupied tables t_{ji} with probability proportional to the number of currently seated customers, or starts a new table t_{j+1} with probability proportional to α. The first customer to sit at a table goes to the buffet line and picks a dish k_{jt} for their table, choosing the dish with probability proportional to the number of times that dish has been picked previously, or ordering a new dish θ_{k+1} with probability proportional to γ.all subsequent customers who sit at that table inherit that dish.

The intuition behind this predictive distribution is that integrating over the global dish probabilities β results in customers making decisions based on the observed popularity of the dishes throughout the entire franchise. That is, dishes are chosen with probability proportional to the number of tables which have previously served that dish [29].

4.2 HDP-HMM

Hierarchical Dirichlet Process Hidden Markov Model introduces the HDP as a prior distribution on infinite dimensional transition matrices. In this way, the number of hidden states of the HMM becomes a variable of the model of which posterior densities can be computed via Gibbs sampling. [4]. The work of Fox [7] further extends the one of Teh and Jordan [30] by introducing an extra variable in the model, the so-called *stickiness* that encourages transitions models with slower changes in the dynamics and enhances self transitions, therefore solving the problem that the original HDP-HMM shows in favoring the learning of models with unrealistically fast-changing dynamics .

Since the HMM involves not a single mixture model, but rather a set of finite mixture distributions, the current state z_t indexes a specific row of the transition matrix, with the probabilities in this row being the mixing proportions for the choice of the *next* state z_{t+1}. The transition probability $\pi(z_t, z_{t+1})$ plays therefore the role of a mixing proportion and the emission distribution F_{z_t} plays the role of the mixture component. Given the next state z_{t+1}, the observation y_{t+1} is drawn from the mixture component indexed by z_{t+1}. To allow for a nonparametric variant of the HMM where the set of states is unbounded, a set of DPs has to be considered, one for each value of the current state. In addition, these DP mixture models must be tied, otherwise the set of states accessible in a given value of the current state will be disjoint from those accessible for some other value of the current state. This yields to a *chain* structure and has the effect that the atoms associated with the state-conditional DPs are shared, which is exactly the framework of the hierarchical DP [30]. The resulting model is thus referred to as the *hierarchical Dirichlet process hidden Markov model* (HDP-HMM).

The HDP-HMM can be expressed as

$$\begin{aligned} z_t | z_{t-1}, \pi_{z_{t-1}} &\sim \pi_{z_{t-1}} \\ y_t | z_t, \theta_{z_t} &\sim F_{\theta z_t} \end{aligned} \tag{4.5}$$

with priors on the parameters and transition probabilities and atoms θ shared across the random base measure G_0. A graphical model representation is shown in fig. 4.1.

A difficulty with the HDP-HMM as mentioned above is that it has a tendency to create redundant states and rapidly switch among them. This can become problematic especially in applications in which the states are the object of inference and when state persistence is expected. This problem can be solved by giving special treatment to self-transitions. In particular, let G_θ denote the transition distribution associated with state θ. In some works by Fox, Sudderth Jordan and Willsky, such as [8] [9] [10], a slightly different definition of this kernel G_θ is proposed:

$$G_\theta|\alpha,\kappa,G_0,\theta \sim DP\left(\alpha+\kappa,\frac{\alpha G_0+\kappa\delta_\theta}{\alpha+\kappa}\right) \tag{4.6}$$

In eq. 4.6, κ represents the *stickiness*, that is an extra-mass placed on the self transition, considering δ_θ the mass point at θ. A stick-breaking representation of the process could be shown in eq. 4.7:

$$\pi_{\theta_k^*}|\alpha,\beta,\kappa \sim DP\left(\alpha+\kappa,\frac{\alpha\beta+\kappa\delta_{\theta_k^*}}{\alpha+\kappa}\right) \tag{4.7}$$

where $\pi_{\theta_k^*}$ is the weight associated with one of the countably states $theta_k^*$ that can be visited by the HMM [29].

Chapter 5

Framework

In this chapter we explains the framework followed in this project, illustrated in figure 1.1. The first section describes the process of obtaining labeled homogeneous segments from a sound, which is based in a previous work by Marchini and Purwins [20]. In the second section, we give a brief overview of the further steps taken to use the HDPHMM system with these segments.

5.1 Homogeneous segments

As a starting point for our work, we examined in detail a previous work by Marchini and Purwins [20]. As said in the introduction, they made an unsupervised system to learn the structure of a rhythmical percussion recording and synthesize musical variations from it. From their system, we used the analysis part, with which we obtain segments of a similar duration and are labeled according the instrument they contain.

In this section this analysis part is summarized, for further details, [20] and [19] are suggested. The analysis part is divided into segmentation of the sound, symbolization of the events in it, temporal alignment, and Homogenify.

5.1.1 Segmentation

This work was focused on the prediction on next notes or events, therefore, the audio signal was segmented by the onset of each musical event. Each of those events is characterized by its position in time (an *onset*) and an audio segment which contains the audio signal from the onset position until the next onset. The segmentation process will serve to compare events to each other in order to generate a reduced *score-like representation*.

5.1.2 Symbolization

The goal of the symbolization stage is to assign to each instrument the same symbol. In order to do that, this stage includes feature extraction, feature clustering, sequence structure analysis and temporal alignment.

Feature extraction The Mel Frequency Cepstral Coefficients (MFCCs) are the measure chosen to describe each event. In particular, the MFCCs are computed frame by frame on the first 200ms of the event weighted by the RMS energy of each frame.

Sound Clustering At this step, each event can be seen as a point in the 13-dimensional space created by the MFCCs. Thus, the *single linkage* algorithm uses the Euclidean distance as the measure to discover event clusters.

The distance between each point creates a binary tree representation of point similarities. With this tree, and from the bottom up, the single linkage algorithm performs clustering by setting different height thresholds. Each threshold results in a different number of clusters, therefore multiple symbol sequences of the same audio file are achieved. This way, it is possible to offer the flexibility of multiple interpretations of the same sound file avoiding problems that could arise with polyphonic sounds. Figure 5.1 illustrates an example of these different clustering configurations.

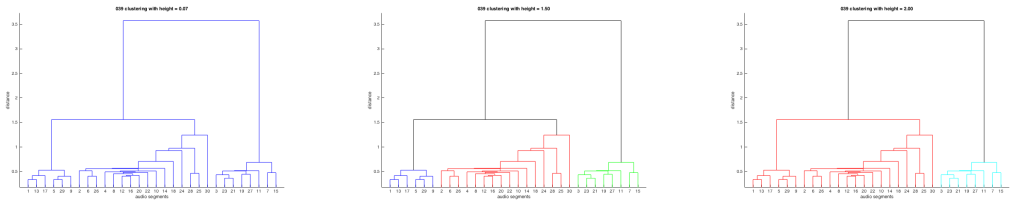


Figure 5.1: Three clustering configurations. Left: 33 clusters. Center: 3 clusters. Right: 2 clusters

At this point, each of these events have been assigned to a cluster.

Level Selection As said previously, the clustering process results in different representations of the audio file. In the *Level Selection* stage, the representation, or clustering level, that best represents the structure of the sound has to be chosen. To find this level, the *regularity*[1] over the levels is computed. The level with the most regularity, the *regular level*, will be used to compute the tempo and metrical grid. However, the levels that contain a local maximum in regularity are kept because a regular interpretation of the sequence is achieved with these levels as well.

[1]Please refer to [19] for further details

5.1.3 Temporal alignment

Beat and tempo alignment The two steps previous to the metrical grid are the *beat* and the *tempo* alignment. In the former, the *skeleton subsequence* [20] is computed. It is based on the symbol with the highest regularity of the regular level. Based on the inter beat interval of this sequence, the tempo of the audio file is computed.

Skeleton Grid With the detected tempo, the preliminary skeleton grid is obtained as a sequence of equally spaced onsets. A further refinement matches the grid points with onsets that are within a tolerance range of $\pm 6\%$ of the it. This procedure regults in a quasi-periodic *skeleton grid* [20].

5.1.4 Homogenization

The last step of the analysis part is the process of homogenization, whose output is a set of labeled audio segments of a similar duration, named *homogeneous segments*. Those that are aligned to an onset, are labeled by the cluster of the aligned onset, otherwise they are labeled by the cluster of the previous homogeneous segment.

In the case when the recorded percussion presents significant tempo deviations, this process can produce artifacts, that is the segments could be mislabeled. In the rest of the chapter, we explain how we have used the ground truth annotations in order to avoid these artifacts when using the HDP-HMM.

It is worth mentioning that this process still is not completely robust. Rhythmic structures with a significant number of syncopated events are not very well analyzed, for instance, and the outcoming homogenized sequences could not describe accurately those structures.

5.2 Path to HDP-HMM

In this section, the rest of the framework is described. First, we explain how the labels for each segment are created. Then, we describe which is the input of the HDP-HMM process.

5.2.1 Matching Homogeneous segments with ENST annotation

As said earlier, we use this framework with audio files taken from the ENST drum database [13]. Along with each audio file, a corresponding annotation file is provided. This file contains a time stamp in second and a label for each drum instrument in the audio file. We have used these annotation to create ground truth labels for each homogeneous audio segment. However, these text files require a bit of processing first.

Making *joint* ground truth annotation files For each time stamp of the annotation file, only one instrument is listed. In the very usual case where more than one instruments are played together, e.g. a Hit Hat and a Bass Drum, the annotation file has two different time stamps (which are very close to each other). This is mainly due to the nature of human playing. Since we assume the onset detector not to be perfect, we have to produce a *joint* annotation file in which the instruments that are supposed to fall on the same metrical position are grouped in the same time stamp. The time difference between the time stamps to be joint has been set to 50 ms.

Some mislabeling artifacts can arise with the Homogenization process, as explained previously. In order to compare the output of the HDP-HMM to a reliable label sequence, a matching between the ground truth labels and the homogeneous labels is required. The result of this matching can be seen in figure 5.2. In the figure, we can see that the last grid point is not aligned to the onset, therefore, the Homogenization process gives to this segment the label of the previous segment.

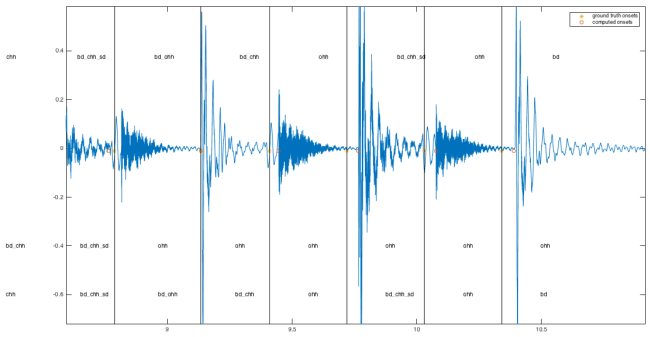


Figure 5.2: Comparison between ground truth labels (at $y = 0.4$), homogeneous labels (at $y = -0.4$) and matching between the two(at $y = -0.6$). The vertical lines represent the metrical grid.

5.2.2 Analysis with HDP-HMM

The aim of the project, as said in chapter 1 is to analyze the unsupervised learning method Hiearchical Dirichlet Process Hidden Markov Models (HDP-HMM), whose theory has been explained in the previous chapters. In this section, the input that has been given to it in order to perform the analysis has been described.

In order to infer a state sequence with HDP-HMM, we have chosen to represent each segment in two ways: with the two first dimensions of the MFCC space and with the two principal dimensions of the MFCC space reduced with Principal Component Analysis. In the next chapter, the results when using these representation along with a different set of parameters will be presented.

Chapter 6

Evaluation

As said before, HDP-HMM is based on the parameters α, γ and the base measure H, which we have fixed as a Gaussian distribution in the evaluation. In particular, the toolbox given by Emily. B. Fox that we have decided to use [6] has, as relevant parameters, the maximum number of states,Kz, the maximum number of Gaussian distributions in each state, Ks, and the concentration parameters, α, and γ[1].

We have decided to evaluate the analysis performance of the HDP-HMM model using two simple sound files from the ENST drum data set. In particular sound files 039 and 042 from drummer library number 2. They both are grouped under the *disco* genre and are classified for their level of complexity. Generally speaking:

- both of these audio files are characterized by a *simple duple* meter, i.e. they can be interpreted either as $\frac{2}{4}$ or $\frac{4}{4}$ time signature;
- both of them contain a regular rhytmic structure based on the repetition of bass drum (BD) and snare drum (SD) on the strong beats, sometimes in conjunction with other drum instruments such as hi-hat (HH);
- both of them are played on similar tempi, and contain hi-hat sounds, either close (CHH) and open (OHH) on *subdivisions*;
- none of them contain tempo or meter changes, as well as syncopations inside the rhythmical structure.

Since these two sound files do not contain more than 10 different instruments, the number of maximum states has been set to 10. Moreover, to simplify the evaluation process, the number of Gaussians in each state has been set to 1. Therefore, the parameters evaluated here are α and γ.

In order to evaluate these parameters, we compare the most repeated state sequence, i.e. clustering index assigned to each sound segment, resulting from the

[1]Please, refer to 4 for details.

HDP-HMM to the state sequence resulting from a matching between the homogenization process and the ground truth labels, as explained in 5.

To find the most repeated state sequence, we have run ten trials of the HDP-HMM inference sampler for a thousand two hundred iterations for a different set of initial parameters. The first 200 iterations are discarded because of the bad approximation that is achieved in that stage, often called the burnout stage.

6.1 Adjusted Rand index

In order to compare clustering results against external criteria, a measure of agreement is needed. We have chosen Adjusted Rand Index (ARI) [31] as measure for how similar the two state sequences are. Doing so, we assume that each segments could be assigned to only one cluster.

The Rand index [27] has been used in several cluster validation studies. As Rand explains, there is no absolute scheme which to measure the evaluation of the performance of a clustering method and comparison of two arbitrarily clusterings is a natural extension of this measurement:

> First, clustering is discrete in the sense that every point is unequivocably assigned to a specific cluster. Second, clusters are defined just as much by those points which they do not contain as by those points which they do contain. Third, all points are of equal importance in the determination of clusterings.

The Rand index is calculated by

$$Rand = \frac{a+d}{a+b+c+d} \tag{6.1}$$

where, given two clusterings P and Q:

- a represents the objects in a pair placed in the same group in both clustering P and Q;
- b represents the objects in a pair placed in the same group in P and in different groups in Q;
- c representing the opposite of b;
- d represents the objects in a pair placed in different groups in both clustering P and Q;

The Rand index lies between 0 and 1, with 1 meaning the two clusterings agree perfectly.

A problem with the Rand index is that the expected value of the Rand index of two random partitions does not take a constant value [31]. Moreover, the Rand Index tends to give quite large values even when clustering methods are in substantial disagreement. The adjusted Rand index (ARI) proposed by Hubert and Arabie [17] improves the existing one by Rand in such way that the two clusterings P and Q are picked at random. They made this adjustment by considering a distribution for assigning points to clusters under the condition that cluster sizes remained unchanged. Mathematics behind it will be omitted since it goes beyond our scope. However, the ARI formula arises from the general form of an index with a constant expected value

$$AdjustedRandIndex = \frac{Randindex - ExpectedRandindex}{MaxRandindex - ExpectedRandindex} \tag{6.2}$$

which is bounded above by 1, and takes the value 0 when the index equals its expected value [31].

6.2 Audio file: 039 phrase disco simple medium sticks

The time signature of this sound is 4/4 form. It contains 4 full bars, and an ending pulse, that is a bass drum (BD) placed on the strong beat of the 5th bar. The homogenization process has detected 33 number of events in this sound file, that is the detected minimum pulse is based on 8th-note subdivisions. On each of these subdivision, a hi-hat (HH) takes place, either open (OHH) or close (CHH). The metrical grid coming from the homogenization step yields to create 8 segments for each bar, so that we have a pattern made of 4 segments repeating twice for each bar. BD and SD take place only on the strong *tatum*s, and in conjunction with BD and SD as well. There are no smaller tempo subdivision in the audio file. Moreover, in the ground truth 35 onsets are detected, so that 35 segments are possible which are grouped in 6 different clusters. The distribution of the segments into clusters can be seen in the figure 6.1. It could be noticed how some detected onset is really close in distance to its previous one, especially in the ending section of the file. That means some of the segments based on the ground truth have a shorter duration compared to the majority of them, and some pre-processing stage should be taken into consideration if we wish to create a *uniform* metrical structure based on that ground truth.

However, the homogenization process comes out with some artifact partly due to the *human-feel* playing. These tempo *fluctuations* make the homogenization step to get confused and mislabel segments.

In the same way, the distribution of segments over the space defined by the first two MFCCs and over the space defined by the two first principal components of the MFCCs can be seen in figures 6.2 and 6.3. From the images, one can see there is

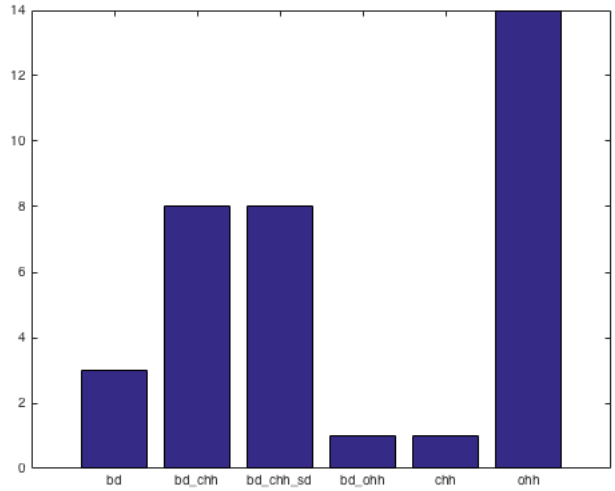


Figure 6.1: Distribution of segments in clusters: 14 of these segments are Open Hit Hat notes (OHH), 8 are Drum with Closed Hit Hat (*BD_CHH*) segments and also Snare Drum with Bass Drum and Closed Hit Hat (*BD_HH_SD*) notes. It follows two segments corresponding to Bass Drum (BD) and, finally, one segment for both Closed Hit Hat (CHH) and Bass Drum Closed Hit Hat (*BD_CHH*).

not that much for difference, nevertheless the best ARI results are achieved when using the PCA projection.

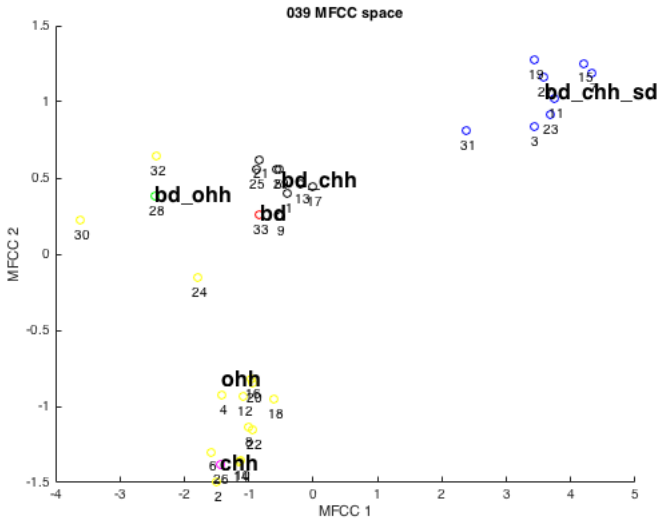


Figure 6.2: MFCC space of 039

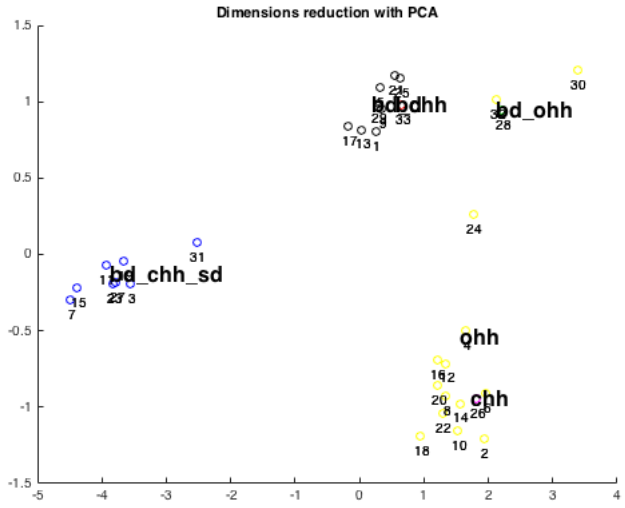


Figure 6.3: First two dimensions of PCA

6.2.1 Evaluation

As said before, we give an ARI value by comparing the most repeated state sequence resulting from the HDPHMM to the state sequence of the ground truth labels. We use the most repeated state sequence

To find the most repeated state sequence, we have run ten trials of the HDP-HMM sampler for a thousand two hundred iterations for a different set of initial parameters of α and γ.

As the results that follow show, the higher the α and γ, the more clusters that should arise from the data.

The higher ARI has been achieved when the estimated clustering had 3 different clusters, in which there were distinguished the segments of BD_CHH (Bass Drum and Hit Hat), BD_CHH_SD (Bass Drum, Hit Hat and Snare Drum), and OHH (Open Hit Hat). The ARI value was of 84%, thus, the segments that were incorrectly classified were the two segments for BD (Bass Drum), the segment for BD_OHH (Bass Drum Open Hit Hat), and the segment for CHH (Closed Hit Hat). Even though this error, the HDPHMM inferred a pattern from the data and assigned to the mislabeled segments to the clusters more usual in that metrical position.

When α is very small, e.g. $\alpha = 0.1$, $\gamma = 1$, the most repeated sequence in the trials assigns all segments in just one cluster, resulting in a 0% ARI. As said before, increasing α results in having more clusters, however, γ needs to be high, as well. In the example, where $\alpha = 1$, $\gamma = 0.1$, the most repeated sequence, with a 38% of ARI, distinguishes between SD segments and the rest. Looking at the MFCCs

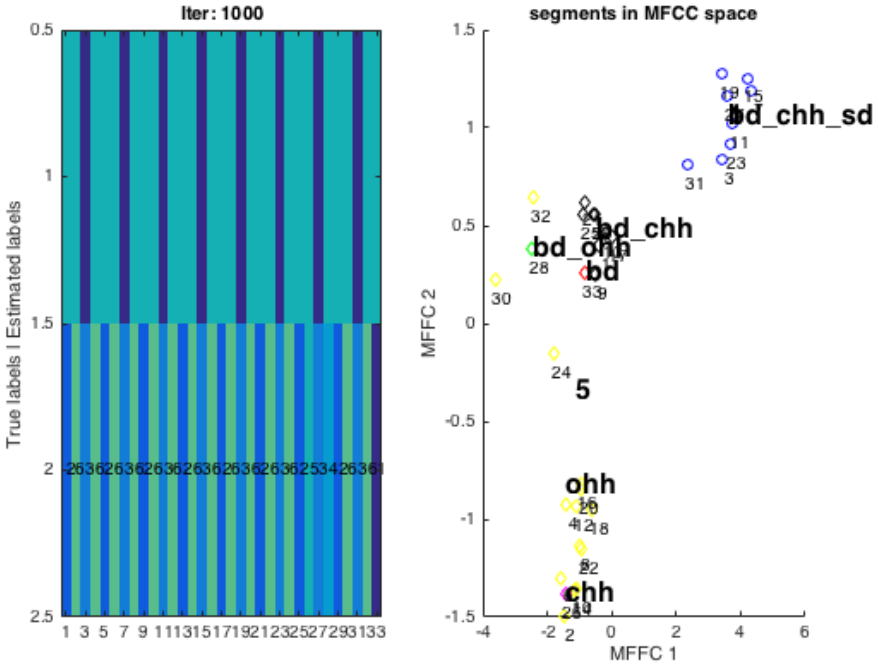


Figure 6.4: Most repeated sequence when using $\alpha = 1$, $\gamma = 0.1$ and MFCC space.
Left: top is the estimated sequence. The bottom sequence is the ground truth sequence. The sequence shows two clusters, one for the snare drum segments and another for the rest of the segments.
Right: scatter plot of the segments. Shape of the points is the estimated sequence and the color is the ground truth label.

scatter plot, one can see that the SD segments are easily separated from the rest. See figure 6.4.

When setting both α and γ parameters to 1, we can see what PCA brings. Using the two first MFCC dimensions, the most repeated state sequence shows two clusters and 38% of ARI. Using the two principal components of the MFCCs, the most repeated sequence has 3 clusters, resulting in a cluster for each family of sounds (BD, SD, and HH), and its ARI is 84%. See figure 6.5.

An interesting case is the case of $\alpha = \gamma = 3$. With this set of parameters, the state sequence presents in all the trials, 4 clusters, with an ARI of 65%. In this case, the metrical position of the HH seems to be taken into account in order to classify these segments as different classes. Therefore, we have a cluster for BD, a cluster for SD, a cluster for HH in beat 2 and a cluster for HH in beat 4. However, the MFCCs plot shows that there is a subtle difference between the HH in second beat that the HH in beat 4, however it is very subtle.

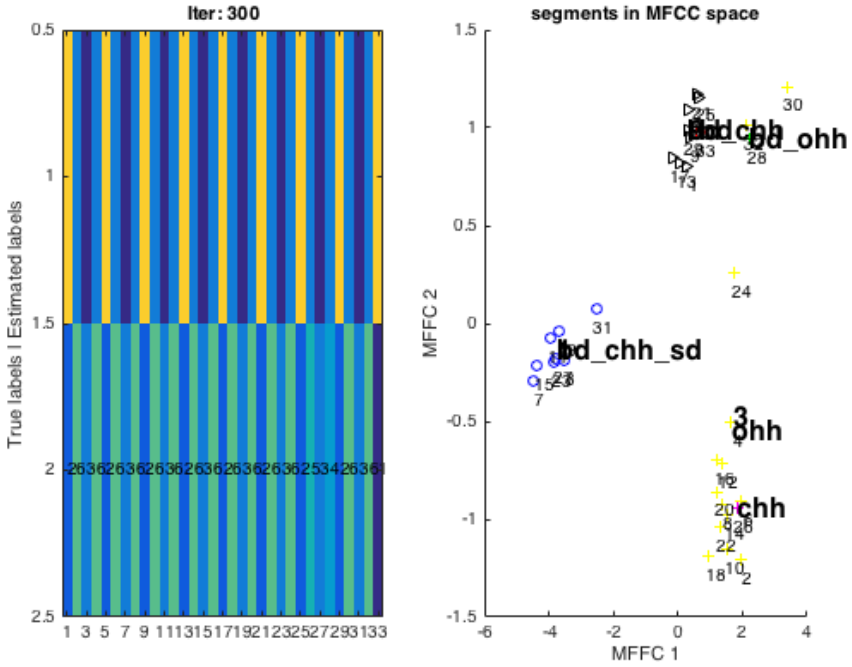


Figure 6.5: Most repeated sequence when using $\alpha = 1$, $\gamma = 0.1$ and PCA space.

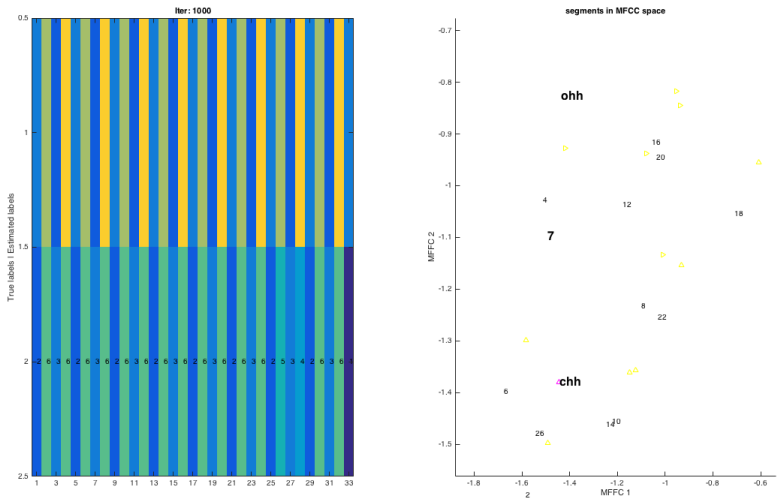


Figure 6.6: Most repeated sequence when using $\alpha = 3$, $\gamma = 0.3$ and MFCC space.
Note the scatter plot in the right is zoomed in the region where the HH are. It shows a different estimated cluster (shape) for HH segments in beat position 2 and 4.

Table 6.1: ARI values for 039

sound	α	γ	ARI 1 (%)	ARI 1 std	ARI 2 (%)	ARI 2 std
039	0.1	1.0	0.08	0.16	0.00	0.00
039	1.0	0.1	0.48	0.27	0.27	0.18
039	1.0	1.0	0.56	0.24	0.66	0.24
039	1.0	1.5	0.52	0.22	0.66	0.24
039	1.0	10.0	0.47	0.19	**0.71**	0.19
039	1.5	1.5	0.56	0.24	0.66	0.21
039	1.5	1.0	**0.68**	0.14	0.68	0.21
039	3.0	3.0	0.65	0.00	0.65	0.00
039	10.0	1.0	0.38	0.00	0.38	0.00
039	10.0	10.0	0.65	0.00	0.65	0.00

ARI 1 represents Adjusted Rand Index using MFCCs, ARI 2 is the ARI using the PCA space.

Figures 6.7 and 6.8 show the whisker diagrams[2] for the ARI values of 10 trials when varying α or γ. From these figures, it is interesting to see that when varying α with a fixed value of γ, the ARI values are not as spread as when γ is varied. We can understand that γ accounts for the variability in the inferred state sequences.

Difference in clustering for different sequences If we extend the clustering investigation to the second and the third most repeated sequences using optimal values for both the parameter, that is $\alpha = 1.5, \gamma = 1.0$, we could notice how the clustering process gives the same total number of cluster, but state sequences slightly differ and so do their ARI measures.

In particular, we could see how the ending part of the audio file has been grouped in different way. For instance, in the second most repeated sequence, segment no. 31 has been clustered as a BD despite being a SD sound and its ARI is 77%. In the third sequence instead, the same segment has been grouped as a SD but the two consecutive audio segments take completely wrong labels, being respectively BD and HH while they should have the opposite labels, that is HH and BD, which have been detected correctly by sequence no. 2. That said, the third sequence has a lower ARI (only 72%) compared to sequence no. 2, and their repetition through the trial doesn't differ so much (39 for the second, 23 for the third). Figure 6.9 shows the plots for both the considered state sequences.

Concluding, this demonstrate how state sequences with the same number of cluster could represent the original structure with several degree of reliability depending

[2]On the Whisker diagrams (or Box plot), the central line is the median, the edges of the box are the 25th and 75th percentiles and the outliers are plotted individually.

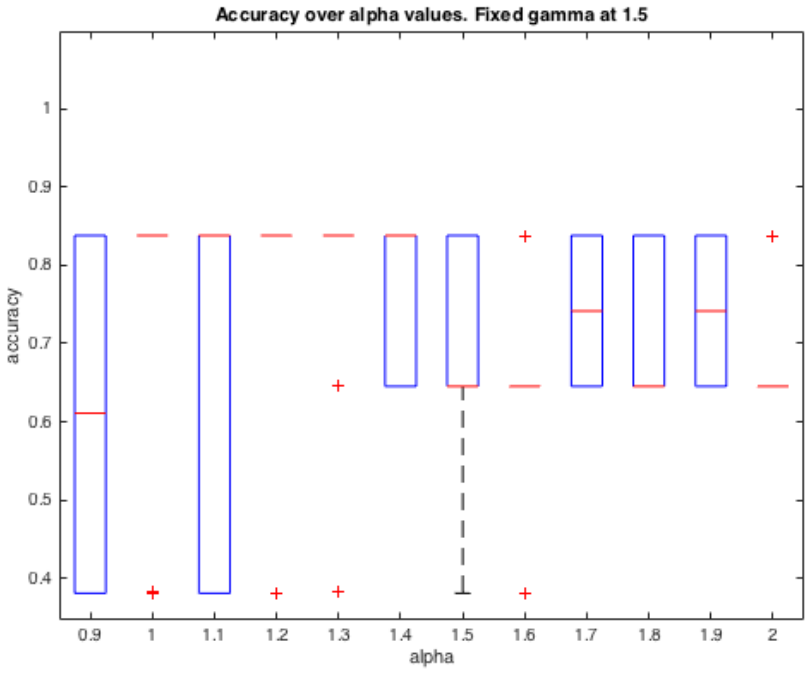


Figure 6.7: Whisker diagram for the ARI when varying α from 0.9 to 2 in steps of 0.1. γ is fixed to 1.5. The value 1.5 has been chosen because good results have been achieved with this value between the two sounds.
In each of the bars, there are 10 ARI values that correspond to the 10 trials run with these pair of parameters.

on how original sounds are clustered.

6.3 Audio file: 042 phrase disco complex medium sticks

Two possible interpretations for this audio file structure could be given. The former yields to a $\frac{4}{4}$ time signature for a total of 4 bars, while the latter yields to a $\frac{2}{4}$ time signature for a total of 8 bars. If we assumed the first interpretation being the most representative, all the bars would have made by two identical halves, with BD and SN falling on the beat and the HHs in their middle, i.e. on the *tatum* level. That said, a $\frac{2}{4}$ time signature representation could fit better this percussive sample. For this audio file, a ground truth of 44 detected onset is provided.

As in the previous example, it is important to point out that the number of onsets goes beyond the expected number of onset, given the time signature, the number of bars and the pulse subdivision. We could therefore experience the same problems as in the former audio file in creating a homogeneous structure based on the ground truth. Moreover, being this percussive pattern a more complex version

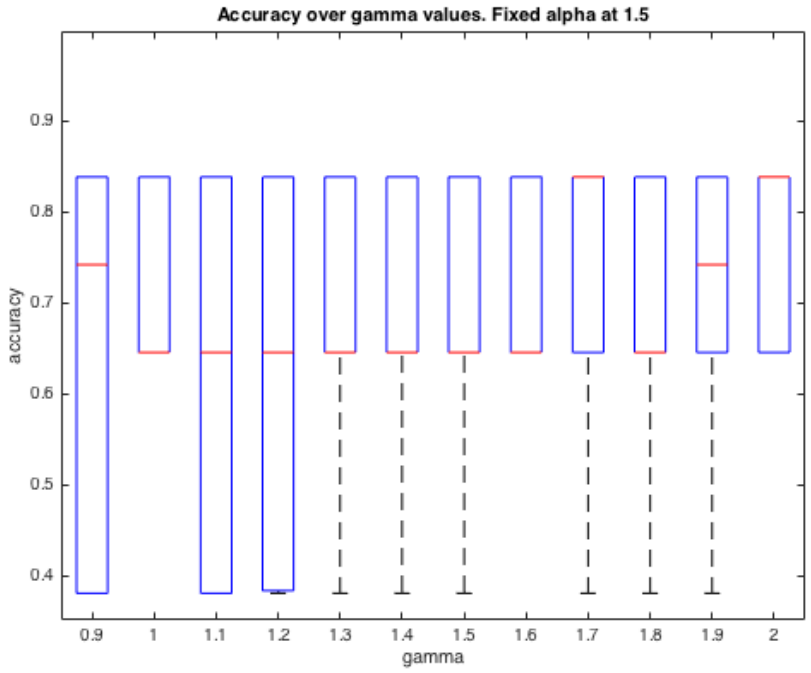
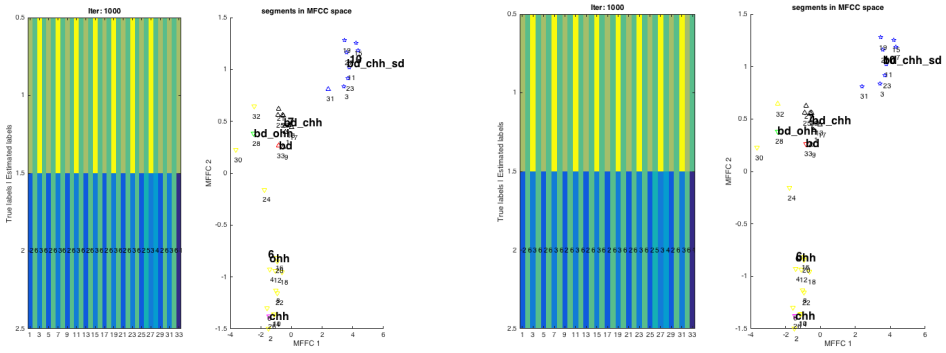


Figure 6.8: Whisker diagram for the ARI when varying γ from 0.9 to 2 in steps of 0.1. α is fixed to 1.5. In the same way as in α, the value 1.5 has been chosen because good results have been achieved with this value between the two sounds.



(a) sequence no. 2 **(b)** sequence no. 3

Figure 6.9: Plot for the second and the third most repeated state sequences for audio file 039

of the previous, we could perceive even 16th-notes subdivision, e.g. a repeated pattern based on such tatum is the combination CHH+OHH before the SD beat. That yields to a reduction in the complexity of the homogenized musical structure where, for instance, that combination is replaced by just one OHH segment and

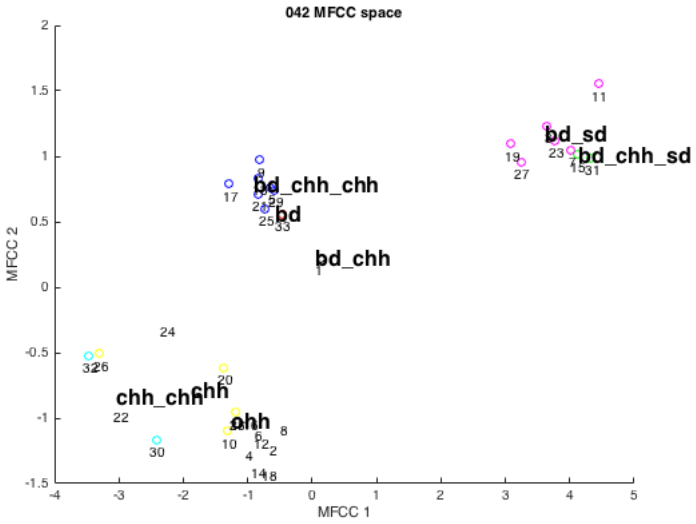


Figure 6.10: MFCC space for 042 audio file

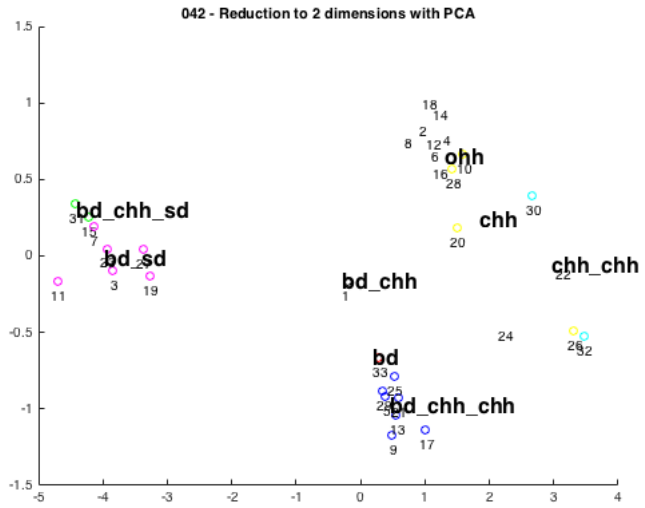


Figure 6.11: First two dimensions of PCA for sound 042 audio file

CHH is merged with the label that comes immediately before, that is BD.

As shown in the precedent example, the distribution over the space defined by the first two MFCCs and by the two principal component of MFCCs is shown in figures 6.12 and 6.11. It is interesting notice how the PCA doesn't help in separating timbres, whereas we can get more defined and separated cluster using only

MFCCs.

6.3.1 Evaluation

As shown previously for the first audio file, we get various results for the overall number of repetitions in the most repeated state sequence as well as for the ARI measures. It can be shown as the Adjusted Rand index measures is always zero for a state sequence made of only one cluster. Such situation is de facto a very unrealistic one, so that a one-cluster state sequence generally yields to a bad representation of the original structure. It happens mostly for much small values of α, e.g. when $\alpha = 0.10$.

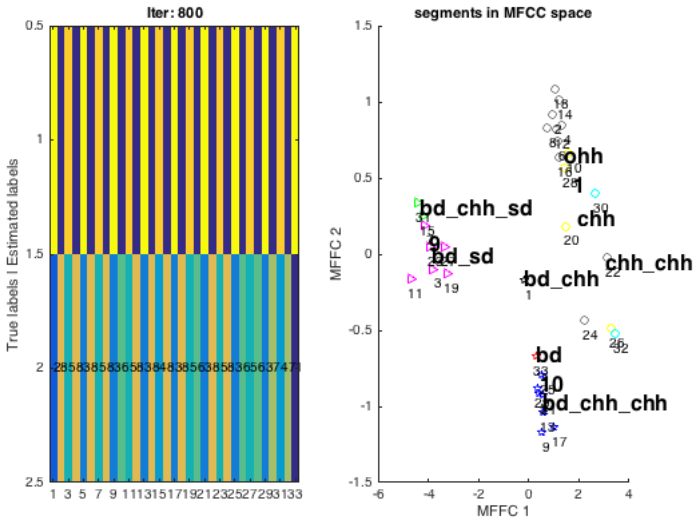


Figure 6.12: Segmentation of the original audio file using $\alpha = \gamma = 1.5$. A three-cluster sequence arises from the inference, whose ARI is 55%.

On the other hand, the higher ARI measure has been obtained for an estimated clustering made up of a three-cluster sequence. In such clustering, a good separation can be noticed among BD, SN and HH as shown in fig. **??** and an ARI of 55% is achieved.

Moreover, the role of the PCA in conjunction with unsupervised learning has to be taken into consideration. Regarding the trials with $\alpha = 0.10, \gamma = 1$ we could see how all the trials give a null ARI using the two first MFCC dimensions, while using the two principal components of the Mel-Frequency Cepstral Coefficients we get two trials with a higher ARI, that is 22%. For this measure, a two-cluster state sequence has been retrieved from the original structure. It is noticeable how in this configuration SD is well clustered, while BD and HH are grouped together despite

their MFCCs are quite different. However, the majority of the most repeated sequences has a number of repetitions that is more than half the whole number of repetitions of the Gibbs sampler for a trial, yielding to a low variability in the definition of a meaningful structure. PCA lastly seems providing a slight improvement in accuracy.

If we switched the values for the parameters, i.e. $\alpha = 1, \gamma = 0.10$, we could notice an improvement in the clustering result. It is shown, when not using PCA, that 3/10 of the most repeated sequences are made of three cluster and their accuracy is 55%, while the trial with the maximum repetition has a bad accuracy and still is made of only one cluster. With PCA the most repeated sequence is also the one with the highest accuracy instead.

When setting $\alpha = \gamma = 1$, a bad ARI is avoided and no one-cluster sequence is retrieved. The overall repetitions in the three-cluster sequences is quite high and all of them have an ARI of 55%. Here, the differences between clustering process with or without PCA on the MFCCs are subtle, and that could be partly due to a good choice of the initialization parameters. That is, an average value around one for both parameters yields to a correct representation of the incoming musical signal. The main difference can be noticed in the number of trials which we get a good ARI from (7 with PCA, 4 without PCA), whereas the number of repetitions for the most frequent sequences seems varying without any specific behaviour between the two options. Again, the three clustering are representing both BD and SD, while CHH and OHH are grouped together in a unique cluster, with no influence by the metrical position.

If changing γ up to an initial value of 10, this doens't provide a better ARI compared to the previous configuration. It turns out that a higher γ couldn't help in a more meaningful representation of the percussive sound. However, the majority of the most repeated sequences have 3 clusters (8/10) even using only the two first MFCC dimensions and no one-cluster sequence is present. Generally speaking, repetitions are lower than in the previous setting - probably because higher γs deal with more variability during the run of the Gibbs sampler. Repetitions for the most repeated sequences decrease dramatically when using PCA, while the highest accuracy still is 55%. One four-cluster sequence is the most repeated in one trial, but its accuracy is lower compared to all the three-cluster sequences. It is interesting to notice how HH is split into two clusters depending on its position. Therefore, in a four-cluster sequence it could be shown how metrical position is taken into consideration since two clusters appear for two different subdivision, which moreover follow different sound, that is different clusters.

The most intereesting case is when $\alpha = \gamma = 1.5$. That seems an optimal values for both α and γ, which provides a quite good overall accuracy throughout all the trials, even though the highest ARI is 55%. Only three- and four-cluster sequences are present, and three-cluster sequences have mostly high repetitions (more than

half the iterations in most cases). Nevertheless, we cannot appreciate meaningful improvements with PCA neither for the number of repetitions nor for the accuracy of HDP-HMM clustering process, on the contrary for one trial a two-cluster sequence is the most repeated and its ARI falls to 22%.

For high values of α, both accuracy and the reliability of the Gibbs sampler are decreasing. It could be noticeable how our *optimal* three-cluster state sequence is completely missing and replaced by a two-cluster sequence for the majority. Moreover, it is interesting how similar sounds have been clustered in different groups in the four-cluster state sequence, e.g. segments no. 14 and 16 (both of them containing a CHH sound). No three-cluster sequences are present neither using MFCC nor using PCA, and the highest ARI is 46%.

If we increase even γ up to a value of 10, all the state sequences are made of four cluster, and repetitions for the most repeated are quite low. PCA seems not providing any meaningful improvement, but the number of repetitions for the most frequented state sequences are quite larger.

The last interesting case is when $\alpha = \gamma = 3$. Here, all the state sequences are made of four cluster as before, but repetitions for the most repeated are quite high, although the highest ARI still is 46%. These results are homogeneous both using MFCC and using PCA.

Difference in clustering for different sequences If we extend the clustering investigation to the second and the third most repeated sequences using optimal values for both the parameter, that is $\alpha = \gamma = 1.5$, we could notice how clustering differs in grouping segments with a different timbre representation in the MFCC space (in this case, the two first MFCC dimensions have been chosen).

While we don't get any significant difference for clustering in the second most repeated sequence, where BD and HH are grouped together, in the third sequence we obtain a different cluster, which groups BD and SD together. We therefore obtain a structure such as shown in fig. 6.13, where each cluster is equally spaced. This structure allows for a good separation between strong and weak beats, so that its ARI is slightly better than the other two-cluster state sequence where BD and HH are linked.

To sum up, several possible optimal values can be possible for both parameters. Particulalry, an optimal range for α is $1 < \alpha < 3$ while even good results are achieved for $\gamma = 1.5$ and $\gamma = 3$. Three- and four-cluster sequences are the most representative of the original structure according to the ARI. However, it is really interesting to point out how a three-cluster configuration make the HDP-HMM inference be more reliable, but also a four-cluster one could be meaningful for this audio file since we have both OHH and CHH sounds, so that they could

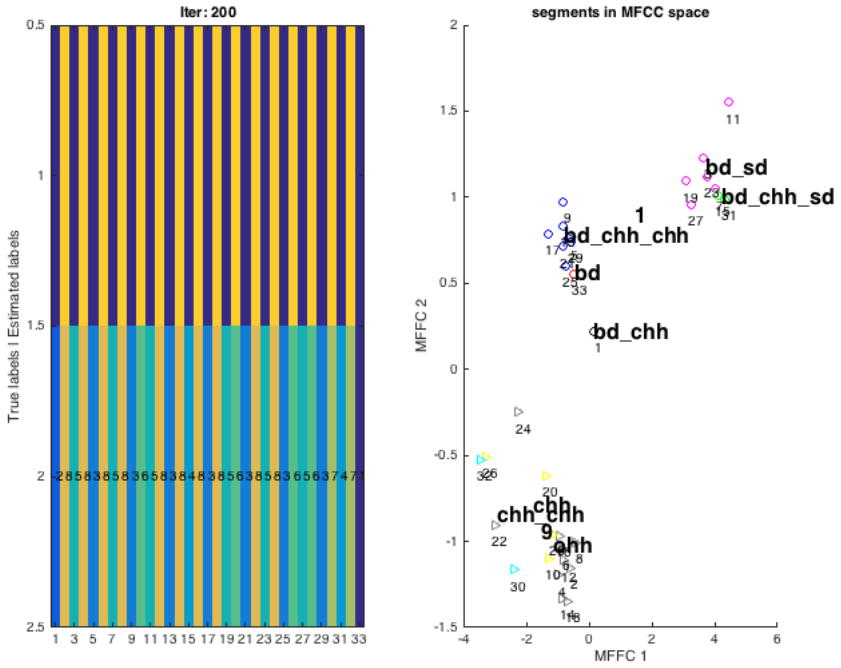


Figure 6.13: Clustering for the third most repeated sequence using optimal values for the parameters $\alpha = \gamma = 1.5$

be grouped into different groups. It is important to underline how the four-cluster sequence doesn't group the HH sounds based on this difference, but it makes a distinction depending merely on their position on the metrical grid. In a possible recombination of those sounds, we couldn't be interested that much either in a such subtle distinction or in such metrical refinement - in fact, we are talking of weak beats. Finally, general good results have been achieved for this percussive sound but ARI shows a lower matching between the sequence coming from the unsupervised learning and the original structure.

Table 6.3: My caption

α	γ	ARI 1 (%) file 039 file 042	ARI 1 std file 039 file 042	ARI 2 (%) file 039 file 042	ARI 2 std file 039 file 042
0.1	1.0	0.08 0.00	0.16 0.00	0.00 0.04	0.00 0.09
1.5	1.5	0.56 0.53	0.24 0.04	0.66 0.49	0.21 0.10
3.0	3.0	0.65 0.46	0.00 0.00	0.65 0.46	0.00 0.00
10.0	1.0	0.38 0.24	0.00 0.08	0.38 0.29	0.00 0.12

Table 6.2: ARI values

sound	α	γ	ARI 1 (%)	ARI 1 std	ARI 2 (%)	ARI 2 std
042	0.1	1.0	0.00	0.00	0.04	0.09
042	1.0	0.1	0.31	0.22	0.42	0.17
042	1.0	1.0	0.35	0.17	0.45	0.16
042	1.0	10.0	0.48	0.14	0.42	0.15
042	1.5	1.5	**0.53**	0.04	0.49	0.10
042	3.0	3.0	0.46	0.00	0.46	0.00
042	10.0	1.0	0.24	0.08	0.29	0.12
042	10.0	10.0	0.46	0.00	0.46	0.00

ARI 1 represents Adjusted Rand Index using MFCCs, ARI 2 is the ARI using the PCA space.

Conclusion In the 039 audio file, we have seen how using the dimensions reduction with PCA improves the ARI values, whereas in the sound 042, the highest ARI is achieved when using the raw MFCC space. In both files, the best ARI values arise when α and γ are between 1.0 and 1.5. We have also seen how the HDP-HMM offers multiple interpretations of the audio file, in which the events are grouped differently. These different interpretations can be useful to segment according to, for example, metrical position or strong-weak beats in order to increase or lower the detail of the representation.

Chapter 7

Conclusion

In this project, an unsupervised system that segments and learns the structure of an incoming audio signal with a novel approach has been presented. This novel approach is the conjunction of the Homogenization process [20] and the Hierarchical Dirichlet Process - Hidden Markov Model (HDP-HMM) statistical model [8]. An overview of the theory of the building blocks of the HDP-HMM has been describes in chapter 3 and 4, as well as the overview of the framework used. Finally, the parameters that take part in the HDP-HMM method have been evaluated using two sounds from the ENST drums database. The project produced a good insight of a recent unsupervised learning method, such as the HDP-HMM. We expect it gives the reader a good sense on the possibilities and how such system works.

Future Works As said in chapter 1, this project is the very first step in a more complex framework that yields to a system which a musician could improvise with. Nowadays many applications using several algorithms and learning methods have been developed, and some of them are already able to create music in a meaningful manner. More and more musicians are interested in applying the concepts of machine learning to music in order to produce a *real* dialogue between performers and machines, that could be adaptive; libraries for musical-oriented programming language such as *ml.lib* [2] and *MnM* [1] represent that will. Unfortunately, fewer of those allow for a online usage [1].
To achieve this online field, one further step could be that of using HDP-HMM even in the generation process instead of a Markov-chain model. In the same way, another further step could be implementing this system in an incremental learning approach. With such an online system, the performer could get a simultaneous feedback from the application using it. In this sense, a *double-sided* improvisation

[1] with the term *online*, we mean a process that, given an instantaneous stimulus, could give back a response based on that stimulus after its analysis. Musically speaking, *online* and *real-time* could be equivalent

could be possible, where both the human performer and the system could affect each other. Moreover, new techniques for the homogenization process could be thought, which yield to the creation of an adaptive metrical grid that takes into account all the possible tempo changes, so to avoid mislabeling or the loss of important information from the original auditory front-end, that is the audio signal in our case.

However, to improve the scope of this project, we should, first, analyze more sounds from the ENST data base. In particular, recordings which contain a more complex rhythm. This would require the onset detection function to be more robust, and, in addition, to have a method to the metrical grid which is robust to syncopated events. It has been shown how homogenization works well when the number of cluster is quite low, but when increasing the variability inside the file, and the rhythmic patterns that it contains, that process starts having dramatic problems. Consequently, they could affect the learning process as well. Results for more complex sounds presented in the ENST database have been omitted, but it could be summarized in short how no meaningful representations of the original structure have been achieved for none of the possible combinations of α and γ parameters.

At the end of this paper, we can observe that all this work is undoubtedly acceptable. Nevertheless, several enhancement and a change for the better are recommended. Finally, the Authors wish for a possible work of revision and improvement during the successive years.

Bibliography

[1] Frédéric Bevilacqua, Rémy Müller, and Norbert Schnell. "MnM: a Max/MSP mapping toolbox". In: *Proceedings of the 2005 conference on New interfaces for musical expression*. National University of Singapore. 2005, pp. 85–88.

[2] Jamie Bullock and Ali Momeni. "ml. lib: Robust, Cross-platform, Open-source Machine Learning for Max and Pure Data". In: ().

[3] David Cope. *Virtual music: computer synthesis of musical style*. MIT press, 2004.

[4] Enrico Di Lello, Tinne De Laet, and Herman Bruyninckx. "Hierarchical dirichlet process hidden markov models for abnormality detection in robotic assembly". In: *Workshop on Bayesian Nonparametric Models (BNPM) For Reliable Planning And Decision-Making Under Uncertainty, NIPS*. Vol. 2012. 2012.

[5] Emily Fox et al. "Bayesian nonparametric inference of switching dynamic linear models". In: *Signal Processing, IEEE Transactions on* 59.4 (2011), pp. 1569–1585.

[6] Emily Fox et al. "Nonparametric Bayesian learning of switching linear dynamical systems". In: *Advances in Neural Information Processing Systems*. 2009, pp. 457–464.

[7] Emily B Fox. "Bayesian nonparametric learning of complex dynamical phenomena". PhD thesis. Massachusetts Institute of Technology, 2009.

[8] Emily B Fox et al. "A sticky HDP-HMM with application to speaker diarization". In: *The Annals of Applied Statistics* (2011), pp. 1020–1056.

[9] Emily B Fox et al. "An HDP-HMM for systems with state persistence". In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 312–319.

[10] Emily B Fox et al. "Bayesian nonparametric methods for learning Markov switching processes". In: *Signal Processing Magazine, IEEE* 27.6 (2010), pp. 43–54.

[11] Bela A Frigyik, Amol Kapila, and Maya R Gupta. "Introduction to the Dirichlet distribution and related processes". In: *Department of Electrical Engineering, University of Washignton, UWEETR-2010-0006* (2010).

[12] Andrew Gelman et al. *Bayesian data analysis*. Vol. 2. Taylor & Francis, 2014.

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

[14] Amaury Hazan et al. "What/when causal expectation modelling applied to audio signals". In: *Connection Science* 21.2-3 (2009), pp. 119–143.

[15] Lejaren Arthur Hiller and Leonard M Isaacson. *Experimental Music; Composition with an electronic computer*. Greenwood Publishing Group Inc., 1959.

[16] Matthew Hoffman, Perry Cook, and David Blei. "Data-driven recomposition using the hierarchical Dirichlet process hidden Markov model". In: *Proc. International Computer Music Conference*. Citeseer. 2008.

[17] Lawrence Hubert and Phipps Arabie. "Comparing partitions". In: *Journal of classification* 2.1 (1985), pp. 193–218.

[18] Olivier Lartillot and Petri Toiviainen. "A Matlab toolbox for musical feature extraction from audio". In: *International Conference on Digital Audio Effects*. 2007, pp. 237–244.

[19] Marco Marchini. "Unsupervised generation of percussion sequences from a sound example". In: *Master's thesis* (2010).

[20] Marco Marchini and Hendrik Purwins. "Unsupervised analysis and generation of audio percussion sequences". In: *Exploring Music Contents*. Springer, 2010, pp. 205–218.

[21] Richard Marxer and Hendrik Purwins. "Unsupervised incremental online learning and prediction of musical audio signals". In: *IEEE Transactions on Audio, Speech and Language Processing* (2015).

[22] Andrew Moore. "Hidden Markov Models". Lecture slides. URL: `http://www.autonlab.org/tutorials/hmm14.pdf`.

[23] Meinard Müller. *Information retrieval for music and motion*. Vol. 2. Springer, 2007.

[24] Richard A. O'Keefe. "An introduction to Hidden Markov Models". Lecture notes. URL: `http://www.cs.otago.ac.nz/cosc348/hmm/hmm.pdf`.

[25] Francois Pachet. "The continuator: Musical interaction with style". In: *Journal of New Music Research* 32.3 (2003), pp. 333–341.

[26] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[27] William M Rand. "Objective criteria for the evaluation of clustering methods". In: *Journal of the American Statistical association* 66.336 (1971), pp. 846–850.

[28] Yee Whye Teh. "Dirichlet process". In: *Encyclopedia of machine learning*. Springer, 2011, pp. 280–287.

[29] Yee Whye Teh and Michael I Jordan. "Hierarchical Bayesian nonparametric models with applications". In: *Bayesian nonparametrics* 1 (2010).

[30] Yee Whye Teh et al. "Hierarchical dirichlet processes". In: *Journal of the american statistical association* (2006).

[31] Ka Yee Yeung and Walter L Ruzzo. "Details of the adjusted Rand index and clustering algorithms, supplement to the paper "An empirical study on principal component analysis for clustering gene expression data"". In: *Bioinformatics* 17.9 (2001), pp. 763–774.