UNIVERSITAT POLITÈCNICA DE VALÈNCIA

# AUDIO CLASSICAL COMPOSER IDENTIFICATION IN MIREX 2015: SUBMISSION BASED ON STRUCTURAL ANALYSIS OF MUSIC

Leopoldo Pla Sempere
*leoplsem@posgrado.upv.es*
Polytechnic University of Valencia

DEPARTAMENT OF COMPUTER SYSTEMS AND COMPUTATION

*Supervised by: Roberto Paredes Palacios*

Master's Degree in Artificial Intelligence, Pattern Recognition and Digital Imaging

M.Sc. Thesis

10th July 2015

*Dedicado a*
*Laura*

# Abstract

This work is the result of interest in neural networks and classical music, knowledge that I wanted to combine after finishing the studies of the IARFID master and professional music degree. I wanted to contribute to an engineering task from the musicological point of view, so, after working on PAN 2015 author profiling task and learning about sentiment analysis, I concluded that I could use similar techniques in MIREX classical composer identification task.

MIREX is the Music Information Retrieval Evaluation eXchange organized by the University of Illinois at Urbana-Champaign (UIUC) in which every year they prepare several Music Information Retrieval tasks held as part of the International Conference on Music Information Retrieval (ISMIR), which this year is celebrated in Malaga. In this exchange exist the Audio Classification (Train/Test) Tasks, and more specifically the Audio Classical Composer Identification, which is perfect for investigating the use of machine learning and music analysis on classical music.

Machine learning is a field of the artificial intelligence which evolves from the study of pattern recognition and include the study of algorithms that can learn and make decisions over data. Specifically, a neural network is a machine learning algorithm that is inspired by the biological neural networks.

Musical analysis is the process whereby a music piece is decomposed and understood as we analyze a text. There is no exact method for these analysis and differs from analyst to analyst. The lowest level of this musical analysis is the harmonic and functional analysis and a higher one is the structure of the segmentation.

This investigation includes the development of the idea and the software that implements the preprocessing and the classification of samples using different software technology layers, as Sonic Annotator, MATLAB and Bash scripts.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

In the last years, Music Information Retrieval fields have been improving step by step and the majority of these fields are still using timbral and chroma features of the audio signal, that is, features based on the timbre of the acoustic wave. On the other hand, in computational linguistics authorship attribution is usual to use features from the text representation as character measures, lexical measures and syntactic features. In fact, one of the features that the winner of PAN 2014 (Evaluation lab on uncovering Plagiarism, Authorship, and Social Software Misuse) in the authorship task[4] and most of author profiling task submissions in 2013[5] use is the part of speech (POS or grammatical) tagging of the text.

So, in this project I approximate the idea of musical structure analysis of classical audio files to identificate composers. This whole concept could be compared to apply POS tagging and segmentation to a speech recording.

Then, in this thesis we will focus on techniques of musical analysis, composition and machine learning instead of signal processing because none of the previous works in the author identification submissions of MIREX used this kind of high level features.

This thesis is distributed in four sections:

- Introduction, in which we establish the state of the art from the MIREX contest and some theoretical concepts about the elements that are involved on the idea of the thesis, as Artificial Neural Networks and

Musical Analysis.

- Implementation, where I explain the used technologies and the different features that I look for in a sample to identify the composer explaining why I chose them and how they work.
- Experiments, which explains and shows the tests with their results under a homemade audio dataset.
- Conclusions, summarizing and expanding the main idea for improvements and useful inferences of the process and results.

## 1.1 MIREX

The Music Information Retrieval Evaluation eXchange (MIREX) edition of 2015 is organized by The International Music Information Retrieval Systems Evaluation Laboratory (IMIRSEL) at the Graduate School of Library and Information Science (GSLIS), University of Illinois at Urbana-Champaign (UIUC). It's hold as part of the 16th International Conference on Music Information Retrieval, ISMIR 2015, which will be held in Malaga, Spain, October 26th-30th, 2015.

This exchange has several kind of tasks related to Music Information Retrieval:

- Grand Challenge on User Experience
- Audio Classification (Train/Test) Tasks, incorporating:
  - Audio US Pop Genre Classification
  - Audio Latin Genre Classification
  - Audio Music Mood Classification
  - Audio Classical Composer Identification
  - Audio K-POP Mood Classification
  - Audio K-POP Genre Classification
- Audio Cover Song Identification

- Audio Tag Classification
- Audio Music Similarity and Retrieval
- Symbolic Melodic Similarity
- Audio Onset Detection
- Audio Key Detection
- Real-time Audio to Score Alignment (a.k.a Score Following)
- Query by Singing/Humming
- Audio Melody Extraction
- Multiple Fundamental Frequency Estimation & Tracking
- Audio Chord Estimation
- Query by Tapping
- Audio Beat Tracking
- Structural Segmentation
- Audio Tempo Estimation
- Discovery of Repeated Themes & Sections
- Audio Downbeat Estimation
- Audio Fingerprinting
- Singing Voice Separation

This work is oriented to the Audio Classification (Train/Test) Task of Audio Classical Composer Identification, which submissions must be

> [...] algorithms to classify music audio according to the composer of the track (drawn from a collection of performances of a variety of classical music genres).[...]

The authors to classify are:

- Bach
- Beethoven
- Brahms
- Chopin
- Dvorak
- Handel
- Haydn
- Mendelssohn
- Mozart
- Schubert
- Vivaldi

## 1.2 State of the Art

Since this work is focused on MIREX exchange, the state of the art is easily obtainable because all the papers and submissions from previous years are stored and classified in the MIREX Wiki[6] and also lots of materials of the ISMIR proceedings[7].

For the specific task of Classical Composer Identification in MIREX 2014 the machine learning algorithms used were Convolutional Neural Networks (CNN)[8] and multiclass Support Vector Machines (SVM)[9][10][11][12], also using PCA[10] and a SVM ranker[9] as a dimensional reduction algorithm. Previous years there were used Restricted Boltzmann Machines (RBM)[13], Gaussian Mixture Models (GMM)[14], k-Nearest Neighbours classifier (kNN) and Multi Layer Perceptron (MLP)[15]. In general, SVM is the most used technique for this task.

From the features point of view, the most used features are timbral features as mel-frequency cepstral coefficients (MFCC)[9][12][14], decorrelated filter banks (DFB)[9][12] and octave-based spectral contrast (OSC)[9][12],

| Algorithm | Normalised Classification Accuracy |
|---|---|
| BK1 | 0.6876 |
| SS6 | 0.6732 |
| SSKS1 | 0.6728 |
| WJ2 | 0.6089 |
| QK2 | 0.5685 |
| BK2 | 0.5213 |
| BK3 | 0.4015 |
| BK4 | 0.3947 |
| XG3 | 0.3312 |
| XG2 | 0.3312 |

Table 1.1: Normalised Classification Accuracies of MIREX 2014.[3]

visual features as spectograms[9], spectral patterns (SP)[16], Delta spectral patterns (DSP)[16] and Short Time Fourier Transform (STFT)[8][10], and also rhythmic patterns as Logarithmic Fluctuation Patterns (LFP)[16] and Correlation Patterns (CP)[16]. After the extraction of these features, sometimes are preprocessed before using them in the machine learning system with some statistical features as mean and variance of the timbral features[9] or with Gabor filters[17], as it is usually done in Biometrics and Computer Vision preprocessing stage.

Note that most of the submissions for Composer Classification task were also submitted for all Audio Classification (Train/Test) Tasks, so they are relatively generic audio classifiers algorithms for music genres, moods and composers.

Also note that none of the submissions is based on any elements of musical analysis.

## 1.3 Neural Networks

One of the most famous techniques of machine learning nowadays is the neural network topology.

The basic definition of a machine learning algorithm like the artificial neural network is a process which an object is represented using acquired

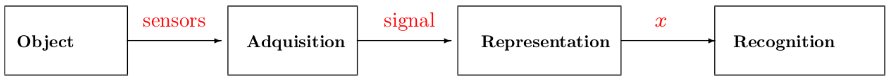

Figure 1.1: Machine learning generic scheme of perception[1]

proprieties from sensors and later is recognized, classified or affects a prediction.

An artificial neural network is a class of classification algorithm that belongs to the connectionist models and it is trained to solve problems by learning the features of the problem through a set of already classified data, or labeled training set (supervised learning). This specific kind of software is designed based on the study of how biologic brain works and it has lots of applications in computer vision, speech recognition, natural language processing, bioinformatics and many other science sectors.

> *The word network in the term 'artificial neural network' refers to the inter-connections between the neurons in the different layers of each system. An example system has three layers. The first layer has input neurons which send data via synapses to the second layer of neurons, and then via more synapses to the third layer of output neurons. More complex systems will have more layers of neurons, some having increased layers of input neurons and output neurons. The synapses store parameters called "weights" that manipulate the data in the calculations. 1.2*
>
> *An ANN is typically defined by three types of parameters:*
>
> - *The interconnection pattern between the different layers of neurons*
> - *The learning process for updating the weights of the interconnections*
> - *The activation function that converts a neuron's weighted input to its output activation. [18]*

Also, this kind of topology is easily paralelizable and it has been implemented in so many frameworks that use several CPU and GPU cores (CUDA or OpenCL, for example).

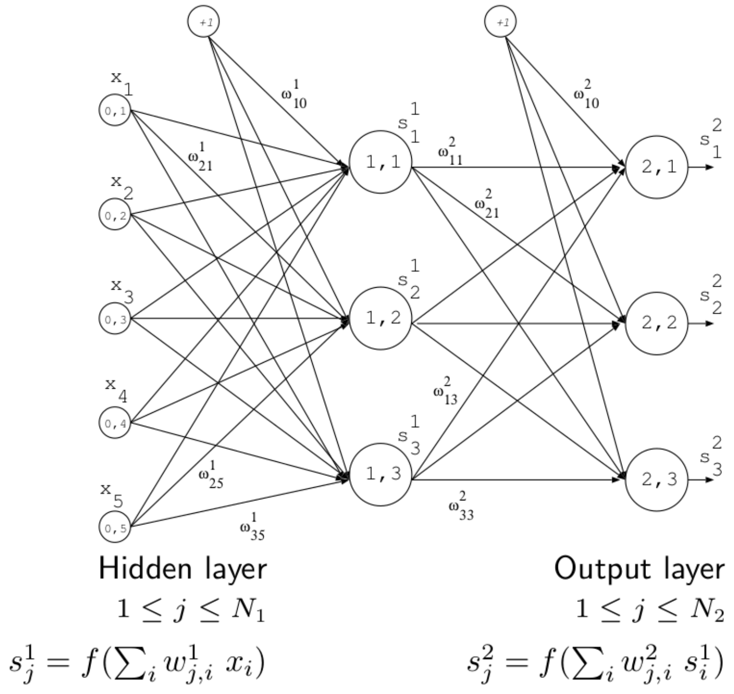

Figure 1.2: General structure of a multilayer perceptron[1]

The origin of the artificial neural networks can be stablished at the first models defined with mathematical notation by McCullock and Pitt in 1943, but the first successful results are obtained in 1959 when the perceptron is created by Rosenblat, which later in 1975 is improved by Paul Werbos adding the backpropagation algorithm.

Some years later the use decreased by the efficiency of Support Vector Machines and some unsupervised learning algorithms due to the difficulty of obtain labeled data.

In the last years, the use of ANN learning algorithms has increased compared to the popular Radial Basis Functions or the Support Vector Machines algorithms, caused by the "boom" of the Deep Learning and the Convolutional Networks.

Nowadays, the most used artificial neural networks are the Restricted Boltzmann Machines, the Convolutional Neural Networks and the Deep Belief Networks.

### 1.3.1 Unsupervised Learning

As Bishop states in his book "Pattern Recognition and Machine Learning":

> *In other pattern recognition problems, the training data consists of a set of input vectors x without any corresponding target values. The goal in such unsupervised learning problems may be to discover groups of similar examples within the data, where it is called clustering, or to determine the distribution of data within the input space, known as density estimation, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization.[19]*

In short, unsupervised learning is a type of learning technique focused on finding hidden structure or drawing inferences from unlabeled input data, as k-Nearest-Neighbours, Gaussian mixture models or Hidden Markov models do. The Restricted Boltzmann machine is a new Deep Learning technique which belongs to the unsupervised learning algorithms and is the main component of Deep Belief Networks.

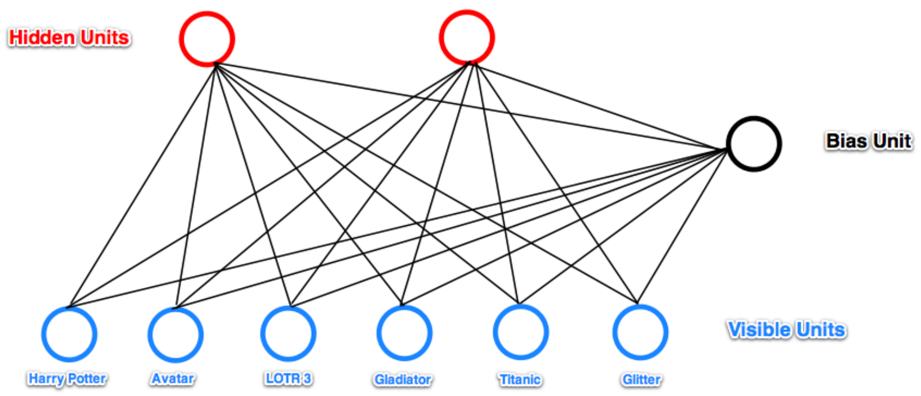

Figure 1.3: RBM used to classify films based on two natural groups: SF-fantasy and Oscar winners[2]

### 1.3.2 Restricted Boltzmann Machines

The Restricted Boltzmann Machines are a type of stochastic artificial forward-backward neural network with sigmoid activation functions. They also are a particular form of log-linear Markov Random Field based on the Energy function.

This neural network is composed by a layer of visible units, a layer of hidden units (the latent factors we try to learn) and a bias unit.

An example of use of a Restricted Boltzmann Machine can be seen in 1.3

### 1.3.3 Deep Belief Network

The Deep Belief Network is a type of Deep Learning algorithms, which is based on a set of stacked Restricted Boltzmann Machines undirectly connected and trained with unsupervised learning, usually the Contrastive Divergence algorithm. These RBMs try to reconstruct the data layer by layer. 1.4 The resultant DBN can be unfolded into a Multi Layer Perceptron (feed forward network) with initialized values and after it can be "fine-tuned" with labeled data using backpropagation.

In the Deep Belief Network

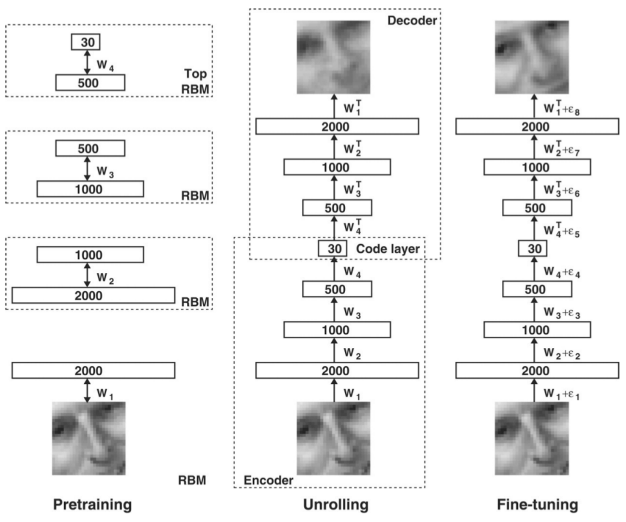

Figure 1.4: Deep Belief Network training example[1]

*[...] the top two layers have undirected, symmetric connections between them and form an associative memory. The lower layers receive top-down, directed connections from the layer above. The states of the units in the lowest layer represent a data vector.*

*The two most significant properties of deep belief nets are:*

- *There is an efficient, layer-by-layer procedure for learning the top-down, generative weights that determine how the variables in one layer depend on the variables in the layer above.*
- *After learning, the values of the latent variables in every layer can be inferred by a single, bottom-up pass that starts with an observed data vector in the bottom layer and uses the generative weights in the reverse direction.[20]*

Since the final result can be unfolded into a MLP, the main practical difference of using DBN is that this resulting MLP is initialized with the reconstructed data of the DBN instead of a random initialization, which usually obtains better results and can be trained with more unlabeled data that the used for MLP.

### 1.3.4 Musical analysis

The process and the academic discipline which studies the musical works from the pattern aspect, the internal structure, composition techniques and interpretative aspects is the musical analysis.

The basic objectives of the musical analysis are the recognition of different elements of the work:

- Structure
- Melody
- Texture
- Harmony
- Tone
- Dynamic

Also, is objective of study the existent relationships inside the previous elements and between them to finally obtain conclusions about the relationships, the compositor ideas and why the composer structured the elements in that way.

Usually, the musical analysis is performed over a score because is easier to identify these mentioned elements, but also listening to some parts of the work is complementary.

The musical analysis includes different subanalysis or techniques that are used to describe the musical work:

- Discretization: breaking the piece down into smaller parts and examine the way these parts are connected. In fact, Fred Lerdahl[21] argues that discretization is indispensable for a musical analysis.
- Harmonic analysis: indicate the harmonic functions on the same score including figured bass and modulations if harmony is tonal.

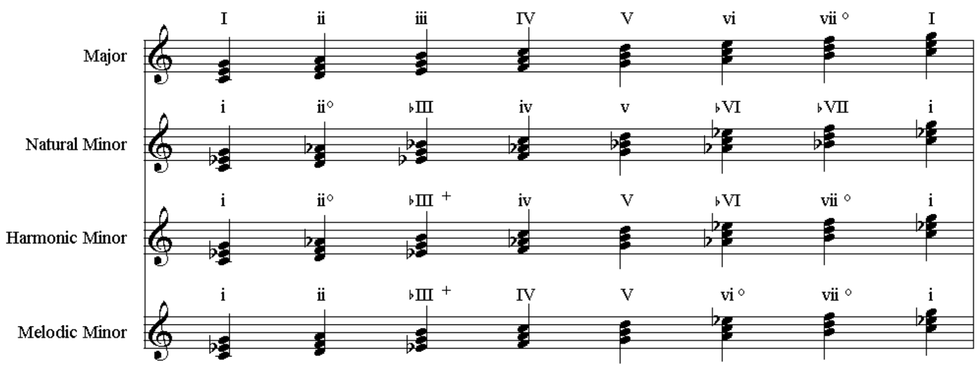

Figure 1.5: Figured bass in different scales of C

Although the discretization is more intuitive to perform, the harmonic analysis implies the knowledge of the harmonies in different scales1.5. Also, the knowledge of the most famous cadences is very important. These are some of them:

- Perfect Authentic Cadence (PAC): V to I (chords in root position)
- Imperfect Authentic Cadence (IAC): V to I (with modified notes)
- Half Cadence (HC): V of V, ii, vi, IV, or I to V
- Plagal Cadence (PC): IV to I
- Interrupted Cadence: V to vi

And there are more cadences and larger common progressions, like "I IV I IV" or "I IV V7 I" that are interesting to obtain from the analyzed work [22].

As an illustrative example of a music analysis, looking at this Beethoven Minuet1.6 we can see the letters above the staff that denotes the different sections of the score. We obtained "a", "b", "c" and again "b". Also, we made the harmonic analysis ("I", "V", "IV" and the other variants, inversions and elements below the staff) focusing on cadences like Half Cadence (HC), Perfect Authentic Cadence (PAC) or Imperfect Authentic Cadence (IAC). There are more elements like passing tones (PT) and subjective descriptions at the margins.

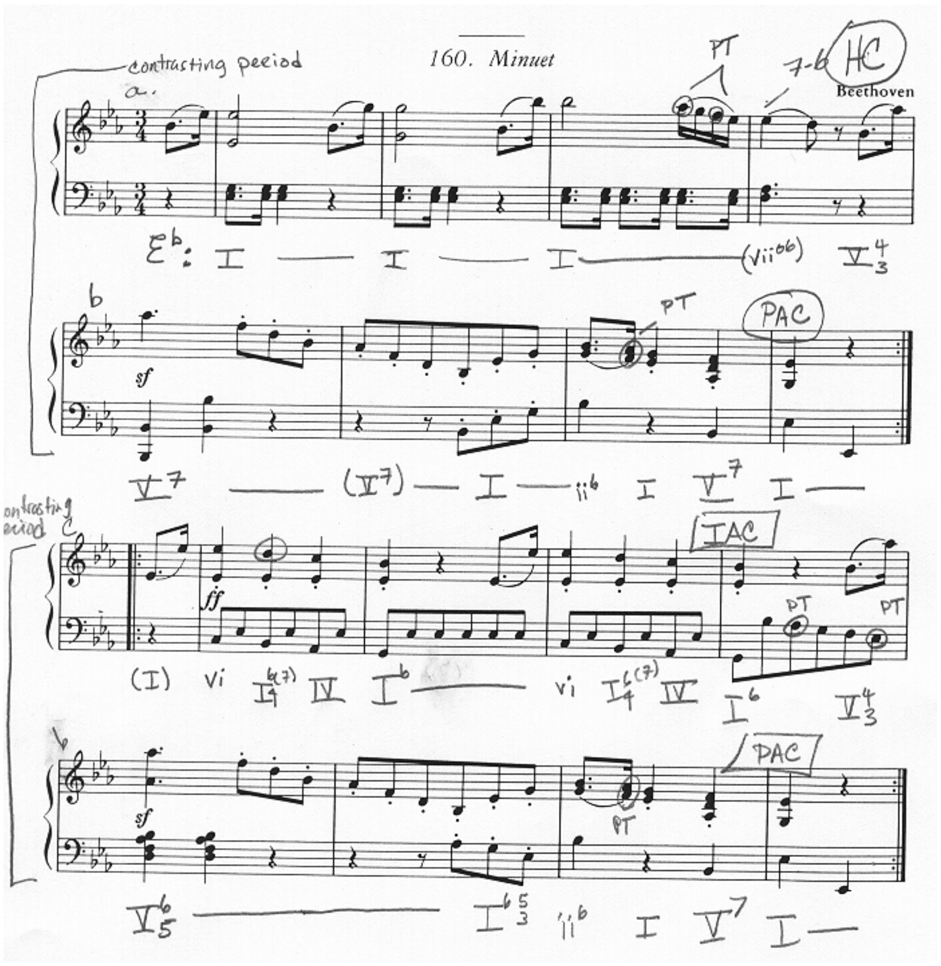
contrasting period
160. Minuet
Beethoven
PT
7-6
HC
E♭: I — I — I — (vii°6) V4 3
PT
PAC
V7 — (V7) — I — ii6 I V7 I —
contrasting period
IAC
PT
PT
(I) vi I6(7) 4 IV I6 — vi I6(7) 4 IV I6 V4 3
PAC
PT
V6 5 — I6 5 3 ii6 I V7 I —
sf
ff

Figure 1.6: Analysis of a Beethoven Minuet No. 3. of 4 in E-flat

But the interesting idea here is that the previous features, like in the texts, help to discriminate the authors.

This field of the music theory covers music from medieval era to the contemporary music, but in this thesis we are focused on classical music analysis and the application of the techniques that we showed in the example.

# Chapter 2

# Implementation

In this section I explain each part of the proposed system 2.5 from the technologies I chose and the features that I extracted to identify the composers.

## 2.1 Technologies

These are the technologies used in all the stages of the system.

### 2.1.1 Sonic Annotator and Visualizer

Sonic Annotator[23] is a batch tool for feature extraction and annotation of audio files using Vamp plugins (binary modules that extract descriptive information from audio data). It was originally developed in 2009 at Queen Mary, University of London as part of the OMRAS2 project, to facilitate feature data publication on the Semantic Web. It has been used in audio annotation projects and it's used to power online audio feature retrieval APIs. It's published under GNU General Public License v2.

Sonic Visualizer[24] is a graphical application that plots the content of the extracted data from Sonic Annotator. It's also developed by the Centre for Digital Music at Queen Mary, University of London and published under

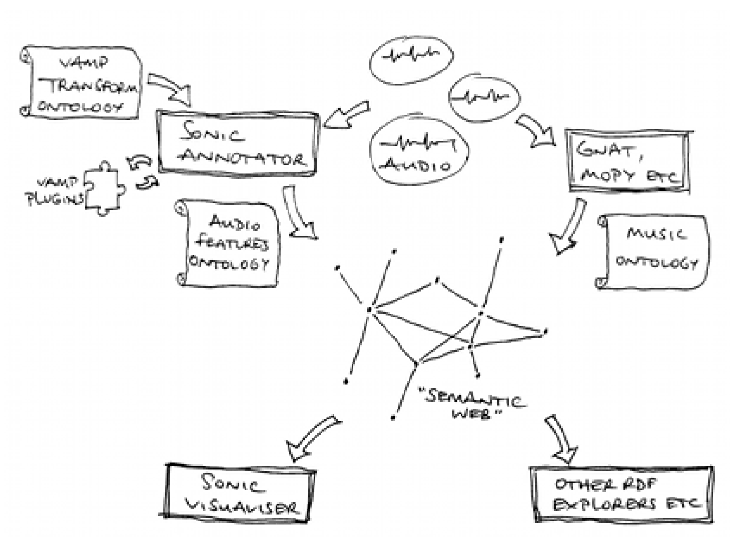

Figure 2.1: Workflow diagram of Sonic Annotator and Visualizer

GPLv2.

Both tools are powerful because they allow you to use the QM Vamp Plugins that were presented to MIREX in previous years[25] and they have an approximate accuracy of 80% in Chord Detection and Key Estimation [26].

### 2.1.2 Python

High level programming language designed to be easy to read, understand and implement. It supports multiple programming paradigms including object-oriented, imperative, functional and procedural. It's published under PSFL licence (open source).

I chose this programming language because it makes easy to work with CSV files and dictionaries (the C++ or Java "hashmaps") for counting the feature values of the audio samples.

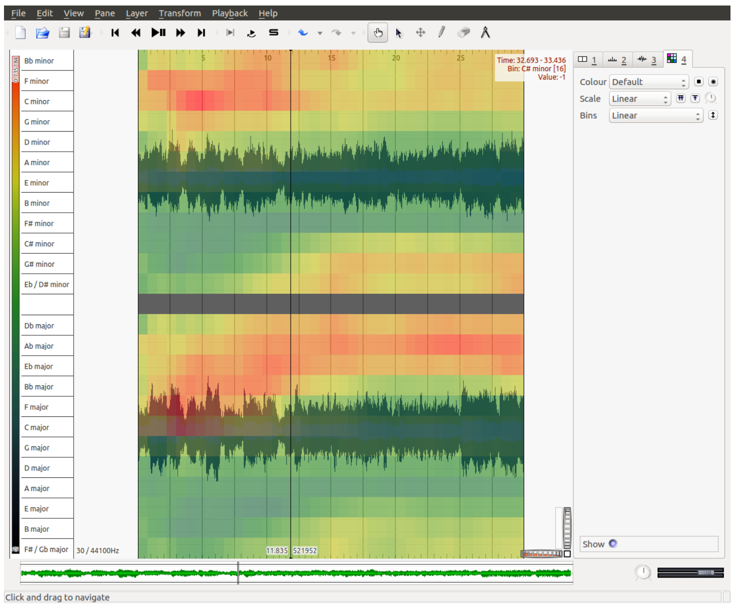

Figure 2.2: Fragment of Bach's Prelude no. 568 with Key Strength Plot in Sonic Visualizer

### 2.1.3 Bash

Bash is a Unix shell and a command language written by Brian Fox for the GNU Project as a free replacement of the Bourne shell. Bash Shell is included in most of GNU/Linux and OS X releases.

The most famous features of Bash (and all Unix shells) are filename globbing (wildcard matching), piping, command substitution, variables and control structures for condition-testing and iteration.

With this features, this tool is perfect to simplify the way of working with thousand of files that need to be passed to other languages scripts or programs, like Python or Sonic Annotator. Also, the example of submission calling formats showed ".sh" scripts for feature extraction, training and classifying samples, so I followed it.

### 2.1.4 MATLAB and Octave

MATLAB is a multi-paradigm numerical computing environment and a proprietary programming language developed by MathWorks. This environment is focused on matrix manipulations and functions and data plotting. For this reasons, this environment (along with Python) is the most used in Mathematics and Statistical fields.

Octave is quite similar to MATLAB and most programs are compatible with both, but Octave is developed as open source and part of GNU Project.

### 2.1.5 Deep Learn Toolbox

The Deep Learning Toolbox of Rasmus Berg is a Matlab/Octave toolbox for Deep Learning. Includes Deep Belief Nets, Stacked Autoencoders, Convolutional Neural Nets, Convolutional Autoencoders and vanilla Neural Nets. It's very easy to use, documented and gives working examples in actual problems giving state of the art results.[27]

Also, given the practice I got in IARFID master using this toolbox in several subjects and the good results and performance it showed, I wanted to save some time from learning more tools using it again.

## 2.2 Feature extraction

In this chapter I explain the features I extracted from audio samples in the training process (at the beginning) to train the neural network and at the end, at the moment to classify new samples.

### 2.2.1 Key Mode

The Vamp Plugin "Key Mode" calculates the major or minor mode of the estimated key in windows of 10 chroma frames. After calculate them, I use

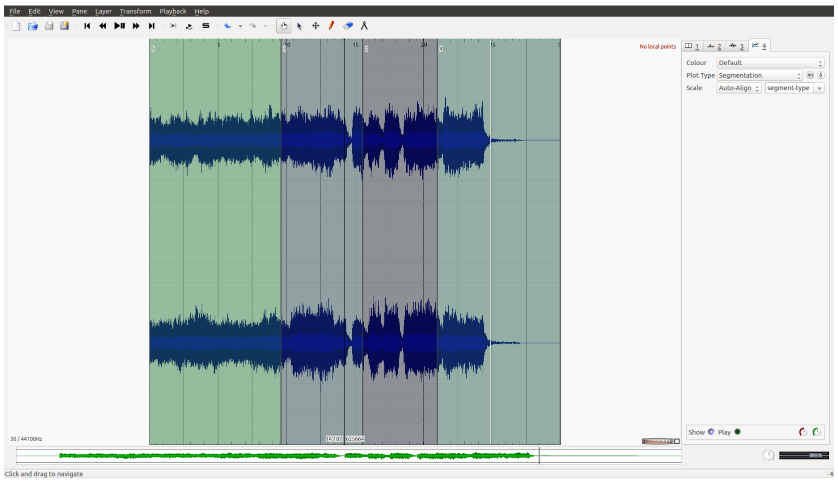

Figure 2.3: Fragment of Bach's Prelude no. 568 with Segmentation plot in Sonic Visualizer

the count of the changes between minor and major as a feature. Major is written as 0 and minor as 1.

### 2.2.2 Segmentation

This is also feature from a Vamp Plugin from Queen Mary which divides the channel into 10 structural segments based on chroma and MFCC. Also, it labels similar segments, which gives us a structural analysis of the sample based on tonality. As key mode, I also use the count of the segments that appear at the segment.2.3

### 2.2.3 Tonality

The main work has been done in these following high level features based on the key of the sample and the detected chords. First of all, I obtain through the key strength Vamp plugin2.2 the value between -1 and 1 of every key (from C, C#, and so on, to B minor and major) of every window of 1 chroma frame. Then, I select the most strong key of every window (in the figure, the most red value of a column) obtaining a vector of keys.

$$Tone(Keys, pond) = argmax_{k \in Keys} \sum_{w_1=1}^{n_{win}} w_k \sum_{w_{10}=2}^{n_{win}} w_k * (1 - w_{(k+7)\%12}) * pond$$

Figure 2.4: Formula to obtain a ponderated key using perfect cadences and key strength

From this keys vector, I created a script in Python 2.4 that obtains the key of the sample using a weighted sum of the number of perfect cadences found at the key strength using 1 chroma and the values of key strength plugin using 10 chroma. In the mathematical notation, keys are sorted by 12 Major tones and then 12 minor tones, the ponderation in the implementation is 12 after some tests and the w is the 1 and 10 chroma window through the algorithm iterates. This obtained key is used as a feature.

### 2.2.4 Harmonic analysis

The previous key vector, then, need to be transposed to convert this incontextual chords into a functional chords to get a functional harmonic analysis of the sample. To transpose, I use the previous obtained key of the sample.

From the previous feature we can obtain the number of functional units in the sample (tonic, dominant, subdominant, etc.), the number of most used cadences (perfect authentic cadence, plagal cadence, half-cadence, etc.) and even a set of most used progressions of three chords (like IV-V-I) 1.3.4. As a parallelism of the POS tagging in text, is interesting to analyze the impact of this features because it's known that discriminate the composers [28].

### 2.2.5 MFCC means

At last, I included the means of the MFCC (Vamp Plugin) from the sample, using 20 coefficients and including C0, also to compare this timbral feature with the structural ones.

## 2.3 Classification

In the proposed systems, I used a Multi Layer Perceptron (Feedforward Back-propagation Neural Network or simply NN) in one system and a Deep Belief Network (DBN) in another one as a classifiers. The features are normalized with z-score to remove outlayers and after that I normalized between 0 and 1 before using the neural network because of artificial neural network input restriction by design. The networks are configured with 44 neurons at the hidden layer, 300 epochs, a batch size based on a divisor of the number of dataset samples, sigmoid activation function and softmax function at the output layer. I tested the system with a homemade database of FLAC files, extracted from my own CD's of the authors of the task.

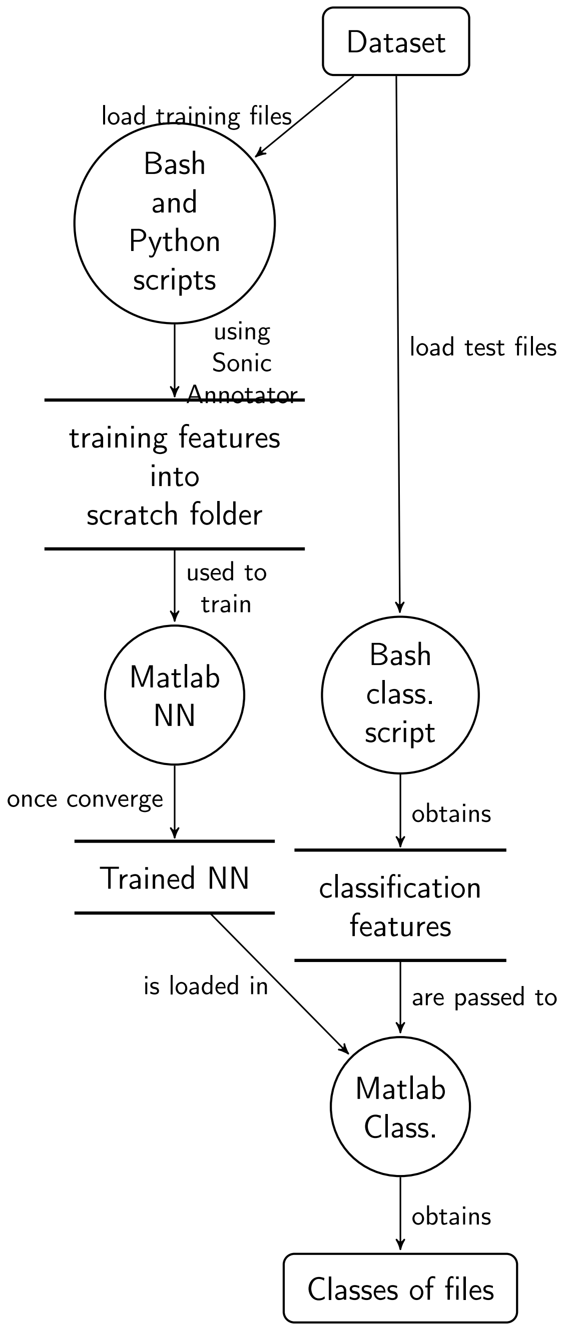

Figure 2.5: Workflow of the proposed system

# Chapter 3

# Experiments

For the experiments, I used a database created by myself. The reason for this decision is that music databases are not easy to distribute because of copyright issues and this task requires a very specific audio samples.

The database I created is structured exactly the same way that the MIREX task specifies:

- 2772 30-second 22.05 kHz mono wav clips
- 11 "classical" composers (252 clips per composer), including:
  - Bach
  - Beethoven
  - Brahms
  - Chopin
  - Dvorak
  - Handel
  - Haydn
  - Mendelssohn
  - Mozart
  - Schubert
  - Vivaldi

The disks I used are (respectively to each composer):

- Bach - The Baroque Music Library (`http://www.baroquemusiclibrary.com/`)
- The Very Best of Beethoven (ASIN: B000B6N64A)
- The Best of Brahms (ASIN: B0000014HA)
- Chopin - Complete Edition (Deutsche Grammophon) (ASIN: B00001X58Z)
- The Very Best of Dvorak (ASIN: B000S5C8O8)
- Handel - The Baroque Music Library (`http://www.baroquemusiclibrary.com/`)
- Haydn - The Baroque Music Library (`http://www.baroquemusiclibrary.com/`)
- Mendelssohn - The complete symphonies (ASIN: B003Y3MYWC)
- Mozart - Complete Works (ASIN: B000BLI3K2)
- Franz Schubert - Masterworks (ASIN: B00062FLJC)
- The Very Best of Vivaldi (ASIN: B000B6N63Q)

After downloading (most of them in MP3) and dumping the audios into WAV files using Asunder app, I created a "quick" bash command to convert all .mp3 files into WAV:

```
find * -type f -name "*.mp3"  -exec avconv -i '{}' '{}.wav'
```

And also another one to cut all files into 30 seconds fragments and classified in folders by composer:

```
for FOLDER in `ls`; do cd $FOLDER; rename 's/^(.{30}).*(\..*)$/$1$2/' * &&
    find | rename 's/[\ \-(),;]//g' && COUNTER=1; for FILE in `ls`; do
    LENGTH=`nohup sox $FILE -n stat | grep "Length␣(seconds):" | sed 's/\ //
    g' | cut -f 2 -d ':' | cut -f 1 -d '.'`; LENGTH=$(( LENGTH - 30 ));
    LENGTH=${LENGTH/#-/}; for i in 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
     18 19 20; do STARTVALUE=`echo $RANDOM % $LENGTH + 1 | bc`; COMMAND="sox
     "$FILE" "${COUNTER}_$FILE" trim $STARTVALUE 30"; COUNTER=$((COUNTER+1))
    ; echo $COMMAND; eval $COMMAND; done ; rm $FILE; done; cd ..; done
```

Also, to test the workflow of the MIREX task, I created the list of files for feature extraction, training and classification with another simple Bash script:

| Features | # of features | DBN | NN |
|---|---|---|---|
| Means MFCC | 20 | 33.5 | **37.1** |
| Means MFCC + Segmentation + Major/Minor | 32 | **27.8** | 25.9 |
| Harmonic Analysis + Means MFCC | 68 | 28.0 | **35.3** |
| Harmonic Analysis | 48 | **12.1** | 10.5 |
| Unigrams+Bigrams + Means MFCC | 45 | 32.3 | **33.6** |
| Unigrams+Bigrams | 25 | **10.1** | 9.4 |
| Unigrams + Means MFCC | 32 | 27.8 | **38.0** |
| Unigrams | 12 | 8.9 | **9.5** |
| Segmentation | 10 | 8.4 | **8.7** |
| Harmonic Analysis + Segmentation + Major/Minor | 60 | **11.3** | 11.1 |
| All | 80 | 18.1 | **29.7** |

Table 3.1: Average three-fold cross-validation accuracies, in percent. Moldface: best performance for a given setting (row).

```
find /home/leopoldo/tfm/audio/ -type f > featureExtractionFiles.txt
cat featureExtractionFiles.txt | cut -f 6 -d '/' > tmp
paste featureExtractionFiles.txt tmp > trainingFiles.txt
find /home/leopoldo/tfm/audio-testing/ -type f > classifyFiles.txt
cat classifyFiles.txt | cut -f 6 -d '/' > tmp
paste classifyFiles.txt tmp > expectedResult.txt
rm tmp
```

This created an amount of 2420 files (6.4GB), but due to the length gap between authors after some tests, I decided to delete some audio fragments from some composers and created a final corpus of 1100 files (100 files per composer).

So, I experimented with my dataset focusing on the harmonic analysis because that features were implemented by myself. I experimented with different Neural Network configurations, with DBN and NN, and using different groups of features to compare the impact of each of them using a three-fold cross validation process.

I named "unigrams" and "bigrams" the figured bass and the cadences given the simile with text n-grams.

NOTE: The results of the MIREX Train/Test tasks are not available at the submission deadline of this report.

# Chapter 4

# Conclusions

In this work we explained a different approach of a music audio classification problem from the music theory point of view using a structural analysis of the musical work. After the results, we can conclude that the propagated error of the feature extraction is crucial for this theory to work. This can be seen at the penultimate test, which uses all the features I implemented and the accuracy is the more or less the same as a "random" classifier, and the MFCC means based classifier gets more than almost all the other results. But we can see little improvements if we compute the classification with figured bass (unigrams).

Also, if we compare with the overall results of MIREX 2014 1.1, the best result is comparable to BK4, which used MFCC, DFB, OSC, Gabor filters, temporal and visual features in a SVM machine[9].

## 4.1 Future work lines

Given this initial idea of working with audio from the harmony and structure of the music, this idea can be improved looking for more plausible patterns to identify:

- Melody or bass patterns (using implementations from the state of the art of MIREX which are close to 90% of accuracy)

- Improve the segmentation feature using history ("n-gram segmentation")
- Other timbral features to compare with (like DFB or OSC)
- More accurate tonality detection to improve the harmony analysis
- Or use different machine learning algorithms like Convolutional Neural Nets

And finally, as this work resulted as a parallelism with PAN contest, this thesis idea can be suggested to be used to investigate the viability of profiling the composer of the classical works to obtain the age of the composer (when the work was composed) or even nationalities due to the different schools of European composers, that can be settle a new interesting task in the MIREX exchange.

# Bibliography

[1] R. P. Palacios, “Redes neuronales artificiales - slides - muiarfid 2015,” 2015.

[2] E. Chen, “Restricted boltzmann machines in python,” 2014.

[3] IMIRSEL, “Summary mirex 14,” 2014.

[4] E. Stamatatos, W. Daelemans, B. Verhoeven, M. Potthast, B. Stein, P. Juola, M. A. Sánchez-Pérez, and A. Barrón-Cedeño, “Overview of the author identification task at pan 2014,” (Bauhaus-Universität Weimar), p. 17, 2014.

[5] F. Rangel, P. Rosso, M. Koppel, E. Stamatatos, and G. Inches, “Overview of the author profiling task at pan 2013,” (Bauhaus-Universität Weimar), pp. 6–10, 2013.

[6] IMIRSEL, “Mirex home,” 2015.

[7] IMIRSEL, “Ismir proceedings,” 2015.

[8] Q. Kong and X. Feng, “Music genre/mood/composer classification: Mirex 2014 submissions,” tech. rep., South China University of Technology, 2014.

[9] S.-R. Baek and M. Y. Kim, “Music genre/mood/composer classification: Mirex 2014 submissions,” tech. rep., Dept. Information and Communication Engineering, Sejong University, Seoul, Korea, 2014.

[10] S. Xu and Y. Gu, “Music genre classification mirex 2014 submissions,” tech. rep., Dept. Electrical and Computer Engineering, University of Rochester, NY, 2014.

[11] M.-J. Wu and J.-S. R. Jang, "Confidence-based late fusion for music genre classification," tech. rep., Computer Science Department, National Tsing Hua University, Hsinchu, Taiwan, 2014.

[12] K. Byun, S.-R. Baek, J.-S. Lee, S.-J. Jang, and M. Y. Kim, "Music genre/mood/composer classification: Mirex 2013 submissions," tech. rep., Dept. of Information and Communication,Engineering, Sejong University, Seoul, Korea, 2013.

[13] A. Pikrakis, "Audio latin music genre classification: a mirex 2013 submission based on a deep learning approach to rhythm modelling," tech. rep., Dept. of Informatics, University of Piraeus, 2013.

[14] F. de Leon and K. Martinez, "Using timbre models for audio classification," tech. rep., University of Southampton, 2013.

[15] P. Hamel, "Multi-timescale pmscs for music audio classification," tech. rep., University of Montreal, 2012.

[16] K. Seyerlehner and M. Schedl, "Mirex 2014: Optimizing the fluctuation pattern extraction process," tech. rep., Dept. of Computational Perception, Johannes Kepler University, Linz, Austria, 2014.

[17] M.-J. Wu and J.-S. R. Jang, "Mirex 2012 submissions - combining acoustic and multi-level visual features for music genre classification," tech. rep., Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, 2012.

[18] Wikipedia, "Artificial neural network — wikipedia, the free encyclopedia," 2015. [Online; accessed 2-July-2015].

[19] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[20] G. E. Hinton, "Deep belief networks," vol. 4, no. 5, p. 5947, 2009.

[21] F. Lerdahl, "Cognitive constraints on compositional systems," *Contemporary Music Review*, vol. 6, no. 2, pp. 97–121, 1992.

[22] J. Craig, "Tonal resources for the creative musician," 2005.

[23] C. Cannam, M. O. Jewell, C. Rhodes, M. Sandler, and M. d'Inverno, "Linked data and you: Bringing music research software into the semantic web," *Journal of New Music Research*, vol. 39, no. 4, pp. 313–325, 2010.

[24] C. Cannam, C. Landone, and M. Sandler, "Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files," in *Proceedings of the ACM Multimedia 2010 International Conference*, (Firenze, Italy), pp. 1467–1468, October 2010.

[25] U. o. L. Centre for Digital Music at Queen Mary, "Qm vamp plugins documentation," 2014.

[26] M. Mauch, K. Noland, and S. Dixon, "Mirex submissions for audio chord detection (no training) and structural segmentation," tech. rep., Queen Mary University of London, Centre for Digital Music, 2009.

[27] R. B. Palm, "Prediction as a candidate for learning deep hierarchical models of data," 2012.

[28] Y. Desportes and A. Bernaud, *Manuel pratique pour l'approche des styles, de Bach à Ravel.* Conservatoire National Supérieur de Paris, 1979.

[29] A. Gkiokas, V. Katsouros, and G. Carayannis, "Ilsp audio music classification algorithm for mirex 2011," tech. rep., National Technical University of Athens, 2011.

[30] J. S. Downie and IMIRSEL, "Mirex 2014 evaluation results," (University of Illinois at Urbana-Champaign), 2014.

[31] P. Magnuson, "Sound patterns: A structural examination of tonality, vocabulary, texture, sonorities, and time organization in western art music," august 2008-2009.

[32] W. T. Berry, *Structural Functions in Music.* General Publishing Company, 1976.

[33] D. Epstein, *Beyond Orpheus: Studies in Musical Structure.* MIT Press Classics, 1979.

# Appendix A

# Donwload links

- Download this document: `https://github.com/Lesbinary/TFM/raw/master/doc/report.pdf`
- Source code download (documentation and implementation): `https://github.com/Lesbinary/tfm/archive/master.zip`
- Github Repository: `https://github.com/Lesbinary/tfm`
- GPLv3 License: `https://github.com/Lesbinary/tfm/blob/master/LICENSE`

# Appendix B

# Environment configuration

This project has been developed under clean install of Ubuntu 15.04 but it also works on Ubuntu 14.04 LTS. The only needed software that is not installed by default is MATLAB and I used version 2014a.

First of all, you need to download my repository (see appendix A) into the folder you want. After that, you need to copy the "vamp" folder of the root of the repository into your home folder ($HOME/vamp) or into /usr/local/lib/vamp because Sonic Annotator searches for plugins in that paths.

Then you are ready to use the script based on the standard of MIREX submissions (`http://www.music-ir.org/mirex/wiki/2015:Audio_Classification_%28Train/Test%29_Tasks#Example_submission_calling_formats`), for example:

```
tfmFeatureExtractor.sh -numThreads 8 ~/tfm/scratch/ ~/tfm/trunk/
    featureExtractionFiles.txt
tfmTrainer.sh -numThreads 8 ~/tfm/scratch/ ~/tfm/trunk/trainingFiles
    .txt
tfmClassifier.sh -numThreads 8 ~/tfm/scratch/ ~/tfm/trunk/
    classifyFiles.txt  ~/tfm/trunk/finalResult
```

And that's it. Also keep in mind that you will need a system with at least 80MB. It's recommended 100-150MB for using a corpus like the one in MIREX. Also keep in mind that there is an option of multithreading which can improve the speed of the run. The default threads value is 4 and can be set with the parameter "-numThreads" (see previous example).