A MUSIC CLASSIFICATION METHOD BASED ON TIMBRAL FEATURES

Thibault Langlois
Faculdade de Ciências da Universidade de Lisboa
tl@di.fc.ul.pt

Gonçalo Marques
Instituto Superior de Engenharia de Lisboa
gmarques@isel.pt

ABSTRACT

This paper describes a method for music classification based solely on the audio contents of the music signal. More specifically, the audio signal is converted into a compact symbolic representation that retains timbral characteristics and accounts for the temporal structure of a music piece. Models that capture the temporal dependencies observed in the symbolic sequences of a set of music pieces are built using a statistical language modeling approach. The proposed method is evaluated on two classification tasks (Music Genre classification and Artist Identification) using publicly available datasets. Finally, a distance measure between music pieces is derived from the method and examples of playlists generated using this distance are given. The proposed method is compared with two alternative approaches which include the use of Hidden Markov Models and a classification scheme that ignores the temporal structure of the sequences of symbols. In both cases the proposed approach outperforms the alternatives.

1. INTRODUCTION

Techniques for managing audio music databases are essential to deal with the rapid growth of digital music distribution and the increasing size of personal music collections. The Music Information Retrieval (MIR) community is well aware that most of the tasks pertaining to audio database management are based on similarity measures between songs [1–4]. A measure of similarity can be used for organizing, browsing, visualizing large music collections. It is a valuable tool for tasks such as mood, genre or artist classification that also can be used in intelligent music recommendation and playlist generation systems.

The approaches found in the literature can roughly be divided in two categories: methods based on metadata and methods based on the analysis of the audio content of the songs. The methods based on metadata have the disadvantage of relying on manual annotation of the music contents which is an expensive and error prone process. Furthermore, these methods limit the range of songs that can be analyzed since they rely on textual information which may not exist. The other approach is based solely on the audio contents of music signals. This is a challenging task mainly due to the fact that there is no clear definition of similarity. Indeed, the notion of similarity as perceived by humans is hard to pinpoint and depends on a series of factors, some dependent on historical and cultural context, others related to perceptual characteristics of sound such as tempo, rhythm or voice qualities.

Various content-based methods for music similarity have been proposed in recent years. Most of them divide the audio signal in short overlapping frames (generally 10-100ms with 50% overlap), and extract a set of features usually related to the spectral representation of the frame. This approach converts each song into a sequence of feature vectors, with a rich dynamic structure. Nevertheless, most of the similarity estimation methods ignore the temporal contents of the music signal. The distribution of the features from one song or a group of songs are modeled, for instance, with the k-means algorithm [3], or with a Gaussian mixture model [1, 5, 6]. To measure similarity, models are compared in a number of ways, such as the Earth-Mover's distance [3], Monte-Carlo sampling [1], or nearest neighbor search. Additionally, some information about the time-dependencies of the audio signal can be incorporated through some statistics of the features over long temporal windows (usually a few seconds), like in [4–8].

In this work we propose computing a measure of similarity between songs based solely on timbral characteristics. We are aware that relying only on timbre to define a music similarity measure is controversial. Human perception of music similarity relies on a much more complex process, albeit timbre plays an important role in it. As pointed out by J.-J. Aucouturier and J. Pachet [1], methods that aim at describing a timbral quality of whole song will tend to find similar pieces that have similar timbres but belong to very different genres of music. For instance, pieces like a *Schumann* sonata or a *Bill Evans* tune will have a high degree of similarity due to their common romantic piano sounds [1]. Following our approach by modeling time dependencies between timbre-based feature vectors, we expect to include some rhythmic aspects in the models. As we will see in section 3.3, this approach leads to playlists with more variety while conserving the same overall mood.

We use a single type of low-level features: the Mel Frequency Cepstral Coefficients (MFCC). The MFCC vectors are commonly used in audio analysis and are described as timbral features because they model the short-time spec-

tral characteristics of the signal onto a psychoacoustic frequency scale. On their own, the MFCC vectors do not explicitly capture the temporal aspects of the music, and therefore are often associated with the "bag of frames" classifiers. In this type of classifiers, songs with the same MFCC frames in different order would be yield the same results. It is our contention that the order of MFCC frames is indeed important and that this information can be used to estimate a similarity measure between songs. We use a language model approach to achieve this result. The most related works include Soltau *et al.* [9], Chen *et al.* [10], and Li and Sleep [11].

In Soltau *et al.* [9], each music is converted into a sequence of distinct music events. Statistics like unigram, bigram, trigram counts are concatenated to form a feature vector that is fed into a neural network for classification. In Chen et al. [10] a text categorization technique is proposed to perform musical genre classification. They build a HMM from the MFCC coefficients using the whole database. The set of symbols is represented by the states of the HMM. Music symbols are tokenized by computing 1 and 2-grams. The set of tokens is reduced using Latent Semantic Indexing. In Li and Sleep, a support vector machine is used as a classifier. The feature are based on n-grams of varying length obtained by a modified version of the Lempel-Ziv algorithm.

This paper is organized as follows: In section 2. we describe our method for music similarity estimation. In section 3. we report and analyze the results of the algorithm on various task and datasets. We also compare performance of our approach to other types of techniques. We close with some final conclusions and future work.

2. PROPOSED APPROACH

The proposed approach is divided into several steps. First, the music signals are converted into a sequence of MFCC vectors [1] . Then, the vectors are quantized using a hierarchical clustering approach. The resulting clusters can be interpreted as codewords in a dictionary. Every song is converted into a sequence of dictionary codewords. Probabilistic models are then built based on codeword transitions of the training data for each music category, and for classification, the model that best fits a given sequence is chosen. The details of each stage are described in the following sections. In the last section we consider building models based on a single music piece, and describe an approach that allows us to define a distance between two music pieces.

2.1 Two-Stage Clustering

The objective of the first step of our algorithm is to identify, for each song, a set of the most representative frames. For each track, the distribution of MFCC vectors is estimated with a gaussian mixture model (GMM) with five gaussians and full covariance matrix (Λ_i):

$$\text{pdf}(f) = \sum_{i=1}^{N} w_i G_i(f) \quad (1)$$

with:

$$G_i(f) = \frac{1}{\sqrt{(2\pi)^d |\Lambda_i|}} \exp\left(-\frac{1}{2}(f-\mu_i)\Lambda_i^{-1}(f-\mu_i)^\top\right) \quad (2)$$

where μ_i represent the Gaussian's mean and f an MFCC frame. We did not perform exhaustive tests in order to chose the optimal value for the number of Gaussians (N) but realized some tests on a reduced number of tracks and decided to use $N = 5$. At this step, the use of GMM is similar to Aucouturier's work [12] were some hints are given about the optimal value of N. The parameters are estimated using the Expectation-Maximization (EM) algorithm. The probabilistic models of the songs are used to select a subset of the most likely MFCC frames in the song. For each track a, $\mathcal{F}_a$, is the set of k_1 frames that maximize the likelihood of the mixture.

Contrasting with Aucouturier's approach, we do not use the GMM as the representation of tracks in the database. This leads to an increased memory requirement during the training phase that is later reduced as we will see in the next section.

The second step consists in finding the most representative timbre vectors in the set of all music pieces. At this stage, the dataset correspond to the frames extracted from each song: $\mathcal{F} = \bigcup_j^{N_m} \mathcal{F}_j$ and the objective is to deduce k_2 vectors that represent this dataset. This is achieved using the k-means algorithm. As an alternative, a GMM trained on the set $\mathcal{F}$ was also used. But thanks to the robustness, scalability and computational effectiveness of the k-means algorithm, better results were obtained using this simpler approach. More precisely, the EM algorithm is sensible to parameters like the number of gaussians and the dimension and the number of data points, and can result in ill-conditioned solutions. That was verified in numerous cases, and we managed to train GMMs with only a reduced number of kernels that was too small for our objectives.

The output of this two-stage clustering procedure is a set of k_2 twelve-dimensional centroids that represent the timbres found in a set of music pieces. The value of the k_1 parameter must be chosen in order to balance between precision [2] , computing and space resources. One of the advantages of dividing into two steps is scalability. Indeed, the first stage has to be done only once and, as we will see in section 3. can be used to compute various kinds of models.

2.2 Language Model Estimation

The set of k_2 vectors obtained during the previous step is used to form a "dictionary" that allow us to transform a track into a sequence of symbols. For each MFCC frame f a symbol s corresponding to the nearest centroid c_i is assigned:

$$s = \underset{i=1..k_2}{\text{argmin}}\ d(f, c_i)$$

[1] Twelve Mel Frequency Cepstral Coefficients are calculated for each frame, all audio files were sampled at 22050Hz, mono and each frame has a duration of 93ms with 50% overlap

[2] We expect that higher values of k_1 parameter will lead to a more accurate description of the set of timbres present in a song.

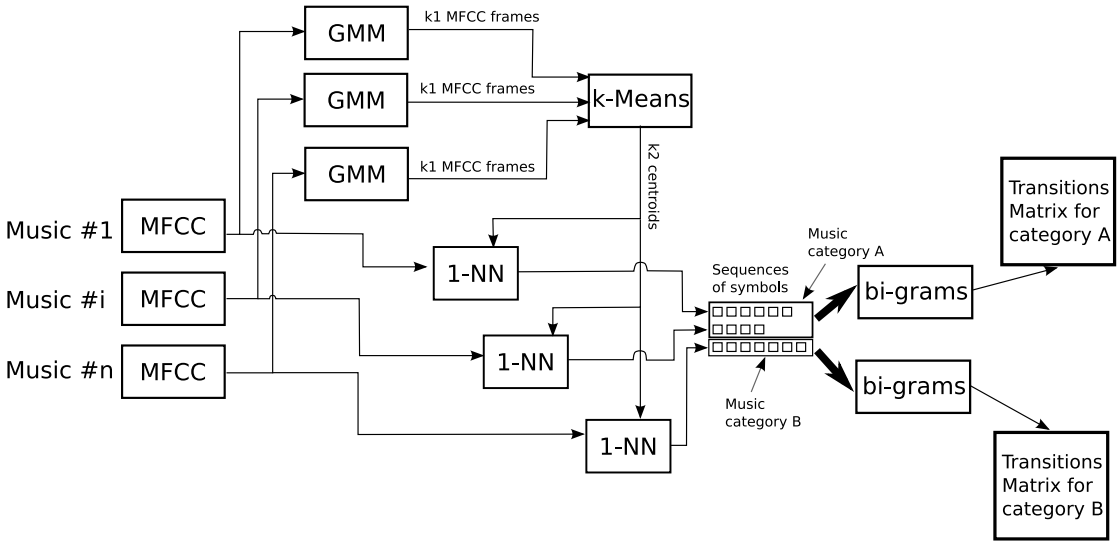

Figure 1. System structure for the language modeling approach. The music signals are converted into a sequence of MFCC vectors, and a two-stage clustering is performed on all the training sequences. Then all the MFFCs are vector quantized resulting in a sequences of symbols. The sequences are divided by category, and the bigrams probabilities are estimated.

where $d()$ is the Euclidian distance. Once tracks are transformed into sequences of symbols, a language modeling approach is used to build classifiers. A Markov Model is built for each category by computing the transition probabilities (bigrams) for each set of sequences. The result is a probability transition matrix for each category containing, for each pair of symbols (s_i, s_j), the probability $P(s_j|s_i)$ of symbol s_i to be followed by the symbol s_j.

This matrix cannot be used like this because it contains many zero-frequency transitions. Many solutions to this problem have been studied by the Natural Language Processing community. Collectively known as "smoothing" the solution consist in assigning a small probability mass to each unseen event in the training set. In the context of this work we experimented several approaches such as the Expected Likelihood Estimator and the Good-Turing estimator [13]. Neither of these approaches are suitable for our case, because the size of our vocabularies is much smaller than those commonly used in Natural Language Processing. We used a technique inspired by the "add one" strategy that consists in adding one to the counts of events. After some tests, we concluded that adding a small constant $\epsilon = 1.0e-5$ to each zero probability transition allowed us to solve the smoothing problem without adding to much bias toward unseen events.

Once a set of models is built, we are ready to classify new tracks into one of the categories. A new track is first transformed into a sequence of symbols (as explained above). Given a model M, the probability that it would generate the sequence $S = s_1, s_2, ...s_n$ is:

$$P_M(s_{i=1..n}) = P_M(s_1) \prod_{i=2}^{n} P_M(s_i|s_{i-1}) \qquad (3)$$

which is better calculated as

$$\begin{aligned} S_M(s_{i=1..n}) &= \log(P_M(s_{i=1..n})) \\ &= \log(P_M(s_1)) + \sum_{i=2}^{n} \log(P_M(s_i|s_{i-1})) \end{aligned} \qquad (4)$$

This score is computed for each model M and the class corresponding to the model that maximize the score values is assigned to the sequence of symbols. One of the benefits of our method is that once the models are computed, there is no need to have access to the audio files and MFCC features since only the sequences of symbols are used. With vocabulary size between 200 and 300 symbols the space needed to keep this symbolic representation is roughly one byte/frame or 1200 bytes/minute.

2.3 Distance Between Music Pieces

Given a database of music tracks, a vocabulary is build following the steps described in section 2.1. Then, instead of creating a model for each "class" or "genre" a model is built for each track (i.e. a probability transtion matrix). Let $S_a(b)$ be the score of music b given the model of music a (see section 2.2). We can define a distance between music a and music b by:

$$d(a,b) = S_a(a) + S_b(b) - S_a(b) - S_b(a) \qquad (5)$$

This distance is symmetric but it is not a metric distance since $d(a,b) = 0 \Rightarrow a = b$ is not verified. It is a difficult task to evaluate a distance between music pieces since there is no "ground truth". One can examine the neighborhood of a song and verify to what extend the songs found nearby show similarities. In our case, the expected similarities should be relative to timbral characteristics since we are using features that represent the timbre. A common application of distances measures over music pieces is to generate playlists. The user selects a song he likes (the

	C	E	J	M	R	W	%acc.	pre.	rec.
Classical	304	2	0	0	0	14	95.0	0.95	0.95
Electronic	1	96	0	0	10	7	84.2	0.74	0.84
JazzBlues	0	2	16	0	6	2	61.5	1.00	0.62
MetalPunk	0	1	0	24	18	2	53.3	0.89	0.53
RockPop	1	13	0	3	78	7	77.5	0.63	0.77
World	17	15	0	0	12	78	63.9	0.72	0.64

Table 1. Confusion matrix, accuracy, precision and recall for each class of the ISMIR 2004 dataset.

seed song) and the system returns a list of similar songs from the database.

3. EXPERIMENTAL RESULTS AND ANALYSIS

3.1 Genre Classification task

We used the ISMIR 2004 genre classification dataset which is composed of six musical genres with a total of 729 songs for training and 729 songs for test [3] . The method described in sections 2.1 and 2.2 was used to classify this dataset. Table 1 shows the confusion matrix on the test set, classification rate, precision and recall for each class, obtained using parameters $k_1 = 200$ and $k_2 = 300$. The overall accuracy is 81.85% if we weight percentages with the prior probability of each class. These results compare favorably with those obtained with other approaches (see for example [5], 78.78% and [14], 81.71%). As can be seen in the following table, the method is not too sensible to its parameters (k_1 and k_2).

k_1	k_2	accuracy	k_1	k_2	accuracy
100	25	74.90%	200	200	81.07%
200	50	77.37%	200	300	81.89%
100	50	79.70%	200	400	81.48%
100	100	80.93%	300	300	81.76%
100	200	81.34%	300	400	81.07%
100	300	81.76%	300	1000	80.52%

3.2 Artist Identification task

One of our objectives with this task is to assess the performance of our method when models are based on smaller datasets. Indeed, contrasting with genre classification, in the case of Artist Identification, a model is build for each artist. We evaluated our method using two datasets: `artist20` [4] that contains 1412 tracks from 20 artists. Each artist is represented by 6 albums. The second dataset focus on Jazz music and is based on authors' collection. It contains 543 tracks from 17 artists (we will call this dataset `Jazz17`). This dataset is smaller than `artist20` but the interest here is to see if our system is able to distinguish songs that belong to a single genre. The abreviations used for the names of the 17 artists are: DK: Diana Krall, SV: Sarah Vaughan, DE: Duke Ellington, TM: Thelonious Monk, CB: Chet Baker, MD: Miles Davis, CJ: Clifford Jordan, NS: Nina Simone, JC: John Coltrane, FS: Frank Sinatra, LY: Lester Young, OP: Oscar Peterson, EF: Ella Fitzgerald, AD: Anita O'Day, BH: Billie Holliday, AT: Art Tatum and NJ: Norah Jones.

Regarding the `Jazz17` dataset, the results are shown in the following table. For two sets of parameter values (k_1 and k_2) the training and test was repeated ten times and the two last columns show the average accuracy and the corresponding standard deviation observed on the test set.

k_1	k_2	mean	std. dev.
100	100	73.49%	1.75
200	200	74.25%	2.25

Because of the reduced number of albums per artist, 50% of each artist's songs were randomly selected and for training while the other half was used for test. Table 2 contains a confusion matrix obtained with `Jazz17`. As can be seen in the confusion matrix, number of misclassifications occur between songs with strong vocals and are thus understandable.

The results obtained with the `artist20` dataset are shown in the following table. We used two different setups. For rows 1 and 2, 50% of an artist's songs are randomly selected and used for training while the other half is used for testing. In rows 3 and 4 we used the strategy suggested in [15]. For each artist an album is randomly selected for test and the other five albums are used for training.

	k_1	k_2	mean	std. dev.
1	100	100	57.40%	0.74
2	200	200	59.14%	1.49
3	100	200	45.28%	7.27
4	200	200	48.98%	7.96

The results shown in rows 3 and 4 are worse than those obtained by Dan Ellis [15] since his approach leads to 54% accuracy using MFCC features and 57% using MFCC and chroma features.

As we can see, choosing the training and testing sets randomly leads to significantly better results than keeping one album for test. This is due to the "album effect" [16]. These results show that despite the name of the task, it is clear that, at least in our case, the problem solved is not the Artist Identification problem. Indeed, our method aims at classifying songs using models based on timbre. Different albums of the same artist may have very different styles, use different kinds of instruments, sound effects and recording conditions. If a sample of each artist's style is found in the training set, it is more likely that the classifier will recognize a song with similar timbre. If every songs of an album are in the test set, then the accuracy will depend on how close are the mixtures of timbres of this album from those of the training set. This is confirmed by the standard deviation observed with both approaches. When trying to avoid the "album effect" we observe a large variation of performance due to the variation of the datasets. In one of our tests we reached an accuracy of 62.3% but this was due to a favorable combination of albums in the training and test sets.

[3] The distribution of songs along the six genres is: classical: 320; electronic: 115 jazzblues: 26; metalpunk: 45; rockpop: 101; world: 122 for the training and the test set.This data set was used for the Genre Classification contest organized in the context of the International Symposium on Music Information Retrieval - ISMIR 2004 (http://ismir2004.ismir.net).

[4] This dataset is available upon request, see: http://labrosa.ee.columbia.edu/projects/artistid/ .

Notwithstanding these observations the results are interesting. In particular with the `Jazz17` dataset, we can see that the timbre-based classification is quite accurate even with music pieces that belong to the same genre.

3.3 Similarity Estimation task

The good results obtained for the classification of large sets of tracks (Genre classification) and more specific sets (Artist Identification) led us to consider building models based on a single track. In this section some examples of playlists generated using our distance are shown and discussed. From our Jazz music set (see section 3.2), we picked some well-known songs and generated a playlist of 20 most similar songs.

In the first example, the seed song is "Come Away With Me" by Norah Jones. The playlist, shown in table 3, is composed of songs where vocals are the dominant timbre. It is interesting to note that with one exception, the artists that appear in this list are all women. The timbre of Chet Baker's voice is rather high and in sometimes may be confused with a women's voice. However, John Coltrane's "Village Blues" appears as an intruder in this list.

	Dist.	Artist	Song
0	0	N. Jones	Come Away with Me
1	4093	N. Jones	Come Away with Me (other version)
2	10774	D. Krall	Cry Me a River
3	11345	N. Jones	Feelin' the Same Way
4	12212	D. Krall	Guess I'll Hang My Tears Out To Dry
5	12333	J. Coltrane	Village blues
6	13015	D. Krall	Every Time We Say Goodbye
7	13201	D. Krall	The Night we Called it a Day
8	13210	N. Jones	Don't Know Why
9	13401	D. Krall	I Remember You
10	13458	D. Krall	Walk On By
11	13758	D. Krall	I've Grown Accustomed To Your Face
12	13852	S. Vaughan	Prelude to a Kiss
13	13915	D. Krall	Too Marvelous For Words
14	13969	D. Krall	The Boy from Ipanema
15	14099	N. Jones	Lonestar
16	14114	C. Baker	My Funny Valentine
17	14405	D. Krall	The Look of Love
18	14674	N. Jones	Lonestar (other version)
19	15039	D. Krall	Este Seu Olhar

Table 3. Playlist generated from "Come Away With Me"

The playlist generated starting with the seed song "Blue Train" by John Coltrane (Table 4) is characterized by Saxophone solos and trumpet. Excluding the songs from the same album, the songs found in the playlist are performed by Miles Davis, Dizzy Gillespie whose trumpets are assimilated with saxophone and Ella Fitzgerald and Frank Sinatra who are accompanied by a strong set of copper instruments.

3.4 Other Approaches

3.4.1 Using unigrams and bigrams

Our classification method is based on models of bigram probabilities whereas most of previous approaches rely on the classification of frame-based feature vectors or on estimates of statistical moments of those features computed on wider temporal windows. In order to quantify the benefit of taking into account transition probabilities an hybrid approach was implemented. With this approach, the classification of a sequence depends on a linear combination of unigrams and bigrams. If we consider only unigrams, the score of a sequence os symbols $s_{i=1..n}$ is:

$$S'_M(s_{i=1..n}) = \log\left(P_M(s_{i=1..n})\right) = \sum_{i=1}^{n} \log\left(P_M(s_i)\right)$$

Using the score computed for bigrams (see equation 4), a linear combination can be writtem as:

$$S''_M(s_{i=1..n}) = \alpha S'_M(s_{i=1..n}) + (1-\alpha) S_M(s_{i=1..n}) \quad (6)$$

where $\alpha \in [0, 1]$. This approach was experimented on the ISMIR 2004 dataset. The results are shown in the following table:

α	1.0	0.5	0.0
accuracy	71.88%	77.64%	81.89%

When $\alpha = 1$, only unigrams are taken into account whereas $\alpha = 0$ reverts to the case where only bigrams are considered. As we can see in this table, the introduction of unigrams in the classification process in not beneficial. A closer look at unigram probabilities give an explaination to these observations. The following table show, for each class, the number of clusters were the class is most represented, the average probability (and standard deviation) of observing the class M given a symbol s, $(P(M|s_i))$.

	Cl.	El.	JB	MP	RP	Wo.
#C	73	68	4	9	16	30
$P(M\|s)$	0.599	0.503	0.578	0.423	0.409	0.471
std.dev.	0.194	0.162	0.180	0.063	0.103	0.139

One can see that for three classes this average probability is below 0.5 i.e. most symbols represents a mixture of timbres. This explains why unigram probabilities are not a good indicator of the class.

3.4.2 Hidden Markov Models

We implemented another technique commonly used to model time-varying processes, the Hidden Markov Models (HMMs). These models were tested on the genre classification task with the ISMIR 2004 genre dataset. The same (discrete) sequences used to train the language models were also used

	Dist.	Artist	Song
0	0	J. Coltrane	Blue Train
1	11367	J. Coltrane	Moment's Notice
2	14422	J. Coltrane	Lazy Bird
3	17344	J. Coltrane	Locomotion
4	23418	E. Fitzgerald	It Ain't Necessarily So
5	25006	E. Fitzgerald	I Got Plenty o' Nuttin'
6	25818	F. Sinatra	I've Got You Under My Skin
7	27054	M. Davis	So What
8	27510	M. Davis	Freddie Freeloader
9	28230	E. Fitzgerald	Woman is a Sometime Thing
10	28598	S. Vaughan	Jim
11	28756	F. Sinatra	Pennies From Heaven
12	29204	D. Gillespie	November Afternoon
13	30299	M. Davis	Bess oh Where's my Bess
14	31796	F. Sinatra	The Way You Look Tonight
15	31971	E. Fitzgerald	There's a Boat Dat's Leavin' Soon for NY
16	32129	E. Fitzgerald	Dream A Little Dream of Me
17	32232	J. Coltrane	I'm Old Fashioned
18	32505	E. Fitzgerald	Basin' Street Blues
19	34045	M. Davis	All Blues

Table 4. Playlist generated from "Blue Train"

	DK	SV	DE	TM	CB	MD	CJ	NS	JC	FS	LY	OP	EF	AD	BH	AT	NJ
DK	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
SV	0	9	0	0	1	0	0	0	0	0	0	1	1	5	0	0	0
DE	0	0	5	0	0	1	0	0	0	0	0	0	0	1	0	0	0
TM	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	0
CB	0	2	0	0	20	1	1	1	0	2	0	0	0	0	0	0	0
MD	0	2	0	0	1	14	1	0	0	1	0	0	0	0	0	0	0
CJ	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0
NS	0	1	0	0	1	0	0	9	0	0	1	0	1	0	0	0	0
JC	0	0	0	0	1	2	0	0	2	0	0	0	1	0	0	0	0
FS	0	0	0	0	0	0	0	0	0	20	0	0	3	0	0	0	0
LY	0	1	0	0	4	0	0	0	0	0	11	2	1	1	1	0	0
OP	0	1	0	0	0	0	0	1	0	0	1	11	0	0	0	0	0
EF	0	4	0	0	0	0	0	0	0	0	2	0	9	2	0	0	0
AD	0	0	0	0	1	0	0	0	0	1	0	0	3	15	0	0	0
BH	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17	0	0
AT	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	18	0
NJ	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16

Table 2. Confusion matrix obtained with the `Jazz17` dataset.

in the HMM's training. For classification, we calculated the probabilities of a given sequence with the HMM's trained for different genres, and assigned the music to the genre with the highest probability.

We used left-right models with 2, 3 and 4 delays, and a fully connected model. We also tested these models with 10 and 20 hidden states. The results, shown in the following table, indicate that the performance of the HMMs is worse than our method. Nevertheless, it should be noted that in our approach, we need a significant number of states (between 100 and 400) in order to achieve reasonable accuracy in timbre modeling. To train an HMM with such a number of hidden states would require a huge amount of data in order for the model to converge.

HMM	LR-2	LR-3	LR-4	FC
10 states	68.3%	69.3%	68.7%	69.1%
20 states	69.1%	69.8%	69.5%	69.5%

4. CONCLUSION AND FUTURE WORK

We described a method [5] for the classification of music signals that consists in a two-stage clustering of MFCC frames followed by a vector quantization and a classification scheme based on language modeling. We verified that the method was suitable for problems with different scales: Genre Classification, Artist Identification and computing of a distance between music pieces. The distance measure, used on a set of songs belonging to a single genre (Jazz), allowed us to derive consistent playlists. The proposed approach was compared with an HMM-based approach and a method that involves a linear combination of unigrams and bigram. On-going work include testing approaches based on compression techniques for symbolic strings.

5. REFERENCES

[1] J.-J. Aucouturier and F. Pachet, "Music similarity measures: What's the use?" in *ISMIR*, France, October 2002.

[2] A. Berenzweig, B. Logan, D. Ellis, and B. Whitman, "A large-scale evaluation of acoustic and subjective music similarity measures," *Computer Music Journal*, vol. 28, no. 2, pp. 63–76, 2004.

[5] This work was partially supported by FCT, through the Multi-annual Funding Programme.

[3] B. Logan and A. Salomon, "A music similarity function based on signal analysis," in *ICME*, 2001.

[4] K. West and P. Lamere, "A model-based approach to constructing music similarity functions," *Journal on Advances in Signal Processing*, 2007.

[5] E. Pampalk, A. Flexer, and G. Widmer, "Improvements of audio-based music similarity and genre classification," in *ISMIR*, 2005.

[6] G. Tzanetakis and P. Cook, "Musical genre classification of audio singals," *IEEE Trans. on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[7] T. Lidy and A. Rauber, "Evaluation of feature extractors and psycho-acoustic transformations for music genre classification," in *ISMIR*, 2005, pp. 34–41.

[8] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl, "Aggregate features and AdaBoost for music classification," *Machine Learning*, vol. 65, no. 2-3, pp. 473–484, 2006.

[9] H. Soltau, T. Schultz, M. Westphal, and A. Waibel, "Recognition of music types," in *ICASSP*, 1998.

[10] K. Chen, S. Gao, Y. Zhu, and Q. Sun, "Music genres classification using text categorization method," in *MMSP*, 2006, pp. 221–224.

[11] M. Li and R. Sleep, "A robust approach to sequence classification," in *ICTAI*, 2005.

[12] J.-J. Aucouturier, F. Pachet, and M. Sandler, "The way it sounds: Timbre models for analysis and retrieval of polyphonic music signals," *IEEE Transactions of Multimedia*, no. 6, pp. 1028 – 1035, 2005.

[13] C. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing*. MIT Press, 2002.

[14] P. Annesi, R. Basili, R. Gitto, A. Moschitti, and R. Petitti, "Audio feature engineering for automatic music genre classification," in *RIAO*, Pittsburgh, 2007.

[15] D. Ellis, "Classifying music audio with timbral and chroma features," in *ISMIR*, 2007.

[16] Y. Kim, D. Williamson, and S. Pilli, "Towards understanding and quantifying the "album effect" in artist identification," in *ISMIR*, 2006.