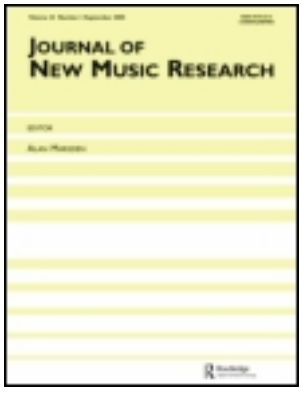

Journal of New Music Research
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/nnmr20

Short-term Feature Space and Music Genre Classification

Gonçalo Marques [a], Thibault Langlois [b], Fabien Gouyon [c], Miguel Lopes [c] & Mohamed Sordo [d]

[a] DEETC-ISEL Lisboa, Portugal
[b] DI-FCUL Lisboa, Portugal
[c] INESC Porto, Portugal
[d] UPF Barcelona, Spain

Available online: 27 Jun 2011

To cite this article: Gonçalo Marques, Thibault Langlois, Fabien Gouyon, Miguel Lopes & Mohamed Sordo (2011): Short-term Feature Space and Music Genre Classification, Journal of New Music Research, 40:2, 127-137

To link to this article: http://dx.doi.org/10.1080/09298215.2011.573563

Journal of New Music Research
2011, Vol. 40, No. 2, pp. 127–137

Routledge
Taylor & Francis Group

Short-term Feature Space and Music Genre Classification

Gonçalo Marques[1], Thibault Langlois[2], Fabien Gouyon[3], Miguel Lopes[3], and Mohamed Sordo[4]

[1]DEETC-ISEL Lisboa, Portugal; [2]DI-FCUL Lisboa, Portugal; [3]INESC Porto, Portugal; [4]UPF Barcelona, Spain

Abstract

In music genre classification, most approaches rely on statistical characteristics of low-level features computed on short audio frames. In these methods, it is implicitly considered that frames carry equally relevant information loads and that either individual frames, or distributions thereof, somehow capture the specificities of each genre. In this paper we study the representation space defined by short-term audio features with respect to class boundaries, and compare different processing techniques to partition this space. These partitions are evaluated in terms of accuracy on two genre classification tasks, with several types of classifiers. Experiments show that a randomized and unsupervised partition of the space, used in conjunction with a Markov Model classifier lead to accuracies comparable to the state of the art. We also show that unsupervised partitions of the space tend to create less hubs.

1. Introduction

In music similarity and genre classification, musical concepts are estimated by models of frame collections and decisions regarding new musical excerpts are made by comparison to these models. It is assumed that at the scale of long-term frame collections, the information contained in the local low-level features is sufficient to infer high-level musical concepts (as e.g. music genres) (Aucouturier, 2006). This is known as the bag of frames approach, where short-term features of the audio signal are considered independent in time, and hence 'put in a bag' without regard for their ordering. Although, there are several attempts to explicitly model temporal information (Soltau, Schultz, Westphal, & Waibel, 1998; Chen, Gao, Zhu, & Sun, 2006; Aucouturier & Pachet, 2007), better results have been achieved by indirectly capturing the signal's time structure through statistics of the short-term features over some time window (e.g. 1 s) (Tzanetakis & Cook, 2002; Lidy & Rauber, 2005; Pampalk, Flexer, & Widmer, 2005; West & Cox, 2005). Nevertheless, this approach to incorporate temporal information only converts a sequence of vectors, each computed on short-term frames of the audio signal, into a shorter sequence of vectors. In the end, decisions are based on global statistics of the feature vectors, disregarding their ordering, either by comparing models of the features' distributions (Logan & Salomon, 2001; Aucouturier & Pachet, 2004; Berenzweig, Logan, Ellis, & Whitman, 2004), or by combining them into one global instance over which standard machine learning techniques are applied (Tzanetakis & Cook, 2002; Pampalk et al., 2005; West & Cox, 2005; Lopes, Gouyon, Silla, & Oliveira, 2010). However, one main assumption remains: starting from a data representation space defined by local signal features, it is implicitly assumed that specific regions of this representation space do *globally* capture musical specificities and, in the context of genre classification, can globally represent genre. Therefore, building good genre models traditionally focuses on determining in an automatic fashion which are these specific representative regions. Once the dimensions (i.e. the low-level features) of the data representation space are defined, the partition of the space and the design of genre models are usually considered jointly.

Throughout our experiments, the data representation space is fixed (local features of audio frames are the same in the whole paper, see Section 2.3). Our objective in this paper is to explore the influence of diverse *partitions* of that space on the performance of a genre classifier. We want to determine if specific regions of the feature space carry relevant musical information about the genres. In

Correspondence: Gonçalo Marques, DEETC-ISEL, Rua Conselheiro Emídio Navarro, 1, 1959-007 Lisbon, Portugal.
E-mail: gmarques@deetc.isel.pt

DOI: 10.1080/09298215.2011.573563

other words, is there a typical set of low-level feature values for a 'e.g. Jazz or Classical' frame? The traditional way of representing tracks using statistics of feature vectors or using parameterization of their distribution is not suitable for our objective. Our method is based on a codebook approach. Although less rich than continuous feature vectors, this vector quantization approach allows an explicit characterization of distinct partitions of the feature space. With this representation, we are able to examine how features from different genres populate the feature space, and how they affect classification results. Furthermore, the same codebook is used to quantize all feature vectors of the training and test datasets, independently of the classes, and prior to building any genre models. This way, we separate the partitioning of data representation space from the process of creating classification models, which are trained only with discrete symbols. In our experiments, we tailor the codebook generation process to favour certain characteristics of the data: we explore the inclusion of more or less information about the origin of the frames used to create the codebook, either by selecting the most representative frames via k-means or via density models of their distributions, or by randomly sampling the data instances. We also explore building codebooks, restricting the training frames to a particular genre. We performed systematic tests with the different codebook generation approaches, and monitored the impact on the classification rate.

Results indicate that short-term feature vectors of *individual* frames are not representative of music genres, and provide additional evidence to previous research advocating that the best way to infer genres from local low-level features is to take decisions at the level of class-dependent *collections* of frames. Yet, even considering frame collections, we observe that there is no apparent benefit in seeking the most thorough representation of each class in the data space defined by local signal features. This is confirmed by experiments in which the codebooks are analysed from an information theoretic perspective: the majority of frames have a large class overlap, while only a few exhibit some level of class dependency, yet, selecting codebook elements based on their class-discriminative capacities does not lead to better classification performance. The results also attest that accuracies on par with current state of the art can be achieved using a randomized and unsupervised quantization of the space in conjunction with Markov-based classifiers, which captures some dynamics of the signal. Finally we compare the effect of similarity measures derived from different codebook generation techniques with a GMM-based similarity measure on hubness. We observe that codebook-based similarity measures tend to create less hubs.

The paper is organized as follows: Section 2 details the codebook generation procedures, the various classifiers used and the datasets. In Section 3, we test various codebook generation strategies, and their influence on the classification rate. Section 4 is dedicated to the study of the clusters (corresponding to the neighbourhoods of codebook elements) from the point of view of information theory. In Section 5, we explore the hubness problem in music similarity tasks. In Section 6, we finalize with a conclusion and suggestions for future directions.

2. Experimental setup

The approach used in this paper consists of using the training dataset to build a codebook. The sequence of frames making up a music piece is then transformed into a sequence of symbols that correspond to the nearest element in the codebook. The following section describes the different strategies used to generate codebooks.

2.1 Codebook generation procedures

A single codebook is used to quantize the set of feature vectors in the training dataset, independently of the class they belong to. The codebook is generated in a two-stage vector quantization approach (Seyerlehner, Widmer, & Knees, 2008; Hoffman, Blei, & Cook, 2009; Langlois & Marques, 2009). First, the feature space is sub-sampled by selecting k_1 vectors from each music piece, and second, k_2 feature vectors are obtained from a set of $N \times k_1$ frames (where N is the number of tracks in the training set). The k_2 feature vectors (clusters) make up the final codebook. For the selection of the k_1 feature vectors, we either:

- select the most representative frames. This is done by building a GMM model for each song (using three kernels and full covariance matrices). The resulting probability density function is used to select the k_1 most representative frames from each track (i.e. the frames that receive highest probability under the fitted GMM);
- select frames randomly according to a uniform distribution. In this case the resulting set of frames will follow the original distribution of data but will also include less representative frames.

For selecting the k_2 elements of the codebook from the $N \times k_1$ set of frames, two procedures are used:

- select the k_2 centroids obtained with the k-means algorithm;
- choose randomly k_2 samples among this set, according to a uniform distribution.

Three combinations of these algorithms are used and are referred to as GMM + k-means (GK), Random + k-means (RK) and Random + Random (RR).

The purpose of the various combinations of algorithms (GK, RK and RR) is to experiment with codebook generation techniques that use different amounts of information from the dataset (respectively less and less). While the GK technique is an attempt to base the codebook on the most representative frames, RR is a technique which, although the resulting codebook follows the distribution of the data, is likely to include less likely representative regions of the space. In this sense, a more extreme method toward less data dependency for codebook generation would be to chose codebook elements on a regular grid, i.e. without taking into account the distribution of the data. Of course this not feasible since our data lie in a 17-dimensional[1] space and in order to collect only two points on each axis we would end up with a codebook of 2^{17} elements. A possible solution is to generate the codebook elements according to a low-discrepancy sequence such as the Sobol sequence. A Sobol sequence generator produces a series of points x_i in a d-dimensional space S^d such that for any integrable function f the series converges as fast as possible and is such that:

$$n \overset{\lim}{\to} \infty \frac{1}{n} \sum_{i=0}^{n} f(x_i) = \int_{S^d} f(x) dx. \tag{1}$$

The objective is to map the space with as few points as possible while minimizing the holes. An interesting property is that if points of a Sobol sequence are projected on a sub-space, they are not superimposed and the holes are minimized. Following this approach, we construct a codebook by generating k_2 points of a Sobol sequence in a 1^{17} hyper-cube and scale these points to fit our data space. The only information used from the dataset is the minimum and maximum values of the feature vectors. We expect these codebooks to be much less efficient in terms of memory because the mapping involves a large portion of the space where there is few or no data. Compared to other approaches, more symbols would be necessary to reach the same level of accuracy. This codebook generation approach is referred to as SS (Sobol Sequences).

2.2 Genre classification

All classification methods used are based on a codebook approach, which implies that each music piece (both for training and testing) is first converted into a sequence of symbols, obtained via vector quantization of the audio features. Classifiers either rely on histograms of the symbol frequency (a histogram is built for every song, each bin indicating the number of times a feature vector was mapped to the corresponding centroid, histograms are normalized to account for different song lengths), or on the temporal dependencies of the symbol sequences.

2.2.1 Histogram + k-NN

In the k-NN algorithm, the music pieces in the training set were used as examples, and a new music piece was classified by a majority vote of its neighbours. In our experiments, we used a 5-NN classifier. The nearest neighbours were calculated based on the Euclidean distance between histograms.

2.2.2 Histogram + SVM

A Support Vector Machine (SVM) (El-Manzalawy & Honavar, 2005) was used with a Radial Basis Function kernel with $\gamma = 1/k_2$ (where k_2 is the number of features, i.e. 200), and a cost $C = 2000$.

2.2.3 Markov models

This classification method is based on the work discribed in Langlois and Marques (2009). The inputs to this classifier are the symbol sequences rather than the songs' histograms. A set of transition matrices is built, one matrix for each genre, containing the probabilities, $P(s_j|s_i)$, of each symbol s_j given the preceding symbol s_i. For classification, the (logarithmic) probability of the test sequence, given each model is calculated:

$$\begin{aligned} \mathcal{L}_M(\mathcal{S}) &= \log\left(P_M\left(s_{i=1,\ldots,n}\right)\right) \\ &= \log(P_M(s_1)) + \sum_{i=2}^{n} \log(P_M(s_i|s_{i-1})), \end{aligned} \tag{2}$$

where P_M represents the symbols probability mass function for the model M. The music class is chosen by the model with the highest score $\mathcal{L}_M$.

2.3 Data sets

We conducted our experiments on two different datasets. The first one is a subset of the Latin Music Database (henceforth, 'LMD dataset'), and the second is the ISMIR 2004 Genre Classification Contest (henceforth, 'ISMIR04 dataset').

2.3.1 LMD

For our experiments, we created a subset of 900 music pieces of the The Latin Music Database (Silla, Koerich, & Kaestner, 2008), which are divided into three groups of equal size (30 pieces per class). The music pieces are uniformly distributed over 10 genres: Axé, Bachata, Bolero, Forró, Gaúcha, Merengue, Pagode, Salsa, Sertaneja, and Tango. We used an artist filter (Pampalk, 2006; Flexer, 2007) so that the music pieces from a

[1]The set of features used is detailed in Section 2.3.

Table 1. Overall accuracies (mean and standar deviation) on the ISMIR04 and the LMD datasets, obtained with different classifiers (lines), and different frame selection techniques (columns).

	ISMIR04 dataset			LMD dataset		
Codebook	GK	RK	RR	GK	RK	RR
Knn (5)	75.53 ± 1.01	75.72 ± 0.81	74.72 ± 1.24	55.98 ± 2.33	58.00 ± 3.14	58.03 ± 2.92
SVM	73.20 ± 0.56	74.98 ± 0.53	74.81 ± 1.29	59.60 ± 1.79	62.41 ± 1.14	62.63 ± 0.87
Markov	81.81 ± 0.38	82.13 ± 0.57	83.03 ± 0.64	69.93 ± 1.40	71.61 ± 1.09	71.58 ± 1.41

specific artist are present in one and only one of the three groups. We also added the constraint of the same number of artists per group.

2.3.2 ISMIR04

This dataset was created for the genre classification contest organized during the ISMIR 2004 conference (Cano et al., 2006).[2] The data is organized into six genres, with a total of 729 music pieces for training and the same number of music pieces for testing. The number of songs per genre is: 320 Classical, 115 Electronic, 26 Jazz-Blues, 45 Metal-Punk, 101 Rock-Pop, and 122 World. We use the same train/test split as in the original ISMIR 2004 contest (with no artist filtering).

2.3.3 Audio features

We extracted 17 audio features from 93 ms frames of the audio signals (mono, sampled at 22,050 Hz, 50% overlap). The features are commonly used in audio genre classification tasks (Tzanetakis & Cook, 2002; Aucouturier & Pachet, 2004; Pampalk, 2006): the zero crossing rate, spectral centroid, rolloff frequency, spectral flux, and 13 MFCCs, including MFCC0.

3. Codebook generation comparison

This section presents results obtained with the various codebook generation techniques described in Section 2.1. Unless specified otherwise, in the remainder of the text, $k_1 = 20$ and $k_2 = 200$. The values of k_1 and k_2 were chosen empirically. The impact of these parameters on the performance of genre classification models was addressed in Langlois and Marques (2009).

3.1 Method comparison

We report the accuracy obtained over test sets only, both for the ISMIR04 and LMD datasets. Every experiment is repeated 10 times, and the performance measure is the accuracy averaged over the 10 test runs.

The evaluation on the LMD dataset first follows a three-fold cross-validation procedure: two groups are used for training and one for testing, with all the permutations of the groups. For the LMD dataset, the experiments were also repeated ten times, and therefore, the reported performances are the average over all 30 runs.

Table 1 shows the average accuracy obtained on both datasets. One can see that, for each classification algorithm (SVM, Knn and Markov), if we consider two standard deviations around the average results (which corresponds approximately to 95% of the area under a normal distribution), there is a strong overlap between the various codebook generation techniques.

When comparing classification methods, we observe that the classifier based on Markov models performs better than the others. Here the main difference is that while the Markov model approach is an attempt to model time dependencies in the sequences of symbols, the other classifiers receive as inputs an estimate of the static distributions of the symbols. Considering the overall accuracy levels obtained with all codebook generation methods (GK, RK, RR), one can see that competitive results can be achieved. For the ISMIR04 data set, the best results found in the literature are around 84% (Pampalk et al., 2005).[3] In the case of the LMD dataset our results cannot be directly compared since we removed some music pieces from the dataset in order to perform artist-filtering (the same artist does not appear in the training and test dataset). In the MIREX 2009 Audio Genre classification contest where the LMD dataset was used, the best result was 74.6%, the average of all participants was 55.5%, and the worst result was 30%. Our results are below the best results but above the average.

We also explore the possibility of defining a codebook via a Sobol sequence, and therefore, without taking into account the distribution of the data. In this context, very large codebooks are necessary to obtain results compar-

[2]http://ismir2004.ismir.net/ISMIR_Contest.html

[3]Some authors (Panagakis & Arce, 2009) have presented results with above 90% accuracy with this dataset but these are obtained through a 10-fold cross-validation procedure and the training set is therefore larger than the one used in our set-up.

able to those obtained with other methods. This is due to the fact that when codebook elements are chosen to better fill the feature space, a large proportion of these elements are in regions of the space were there are very few or no data. Several codebook sizes were experimented with. For each codebook, the evaluation is based on the accuracy obtained with a Markov model based classifier. Figure 1 summarizes the results for codebook sizes ranging from 200 to 40,000. One can see that the accuracy increases with codebook size until 20,000 symbols and then levels off around 81% in the case of the ISMIR04 dataset and around 65% for the LMD dataset.

3.2 Using a subset for codebook generation

Experiments of the previous section showed that there is apparently no clear advantage to model the statistics of the short-term feature vectors and that at this level an unsupervised approach can be considered.

In this section we consider the hypothesis that short-time frame vectors do belong to a class. Several codebooks are built using feature vectors from a single class. The assumption is that if we base the representation on elements of a *single class* and compare the accuracy obtained, we should be able to evaluate the benefit of having a class-dependent representation at this level. In our experiments, codebooks built with frames from only one genre are referred to as 'only-X', while codebooks generated using frames from all classes are referred to as 'all-genres'.

Table 2 shows the accuracy obtained on the ISMIR2004 test set, for codebooks based on one single class, and built using the RR procedure. One can see that for example, using only frames from the Rock-Pop category for the construction of the codebook leads to a decrease of performance of $\approx 1\%$ when comparing to the codebook that is based on all genres (Markov classifier). When looking at the worse case, the difference with the reference codebook is $\approx 3.6\%$ in the case of the codebooks based on Classical or Metal-Punk music. It means that with a representation based solely on Metal-Punk music the classification rate over all genres decreases by only 3.6%. This decrease in accuracy is rather small if we consider the perceived difference between the musical genres. The same experiment was performed with the LMD dataset. The results shown in Table 3 share the same characteristics with, in the majority of cases, a smaller difference ($\approx 1\%$) in the accuracy obtained with the reference codebook. A notable exception is Tango. This result can be explained by the fact that the tango music of the LMD database is composed mainly of old recordings (from 1917–1935) with very poor sound quality. The set of feature vectors extracted from this genre have a limited frequency range and do not account for the variety found in other genres.

3.2.1 Summary

The results of experiments presented in this section permit us to draw some conclusions. (1) Selecting the codebook elements randomly (the RR approach) leads to levels of performance at least as good as those obtained when selecting the most representative frames through Gaussian Mixture Models (the GK approach). (2) Selecting frames from a single genre to generate the codebook does not degrade the performance. (3) It is possible, using Sobol sequences, to build a representation space for music data knowing very little about the data at hand and simultaneously achieve good levels of performance in terms of accuracy. (4) The codebook approach when used in conjunction with Markov model based classifiers allow us to achieve state-of-the-art performance levels on genre classification tasks. (5) There is

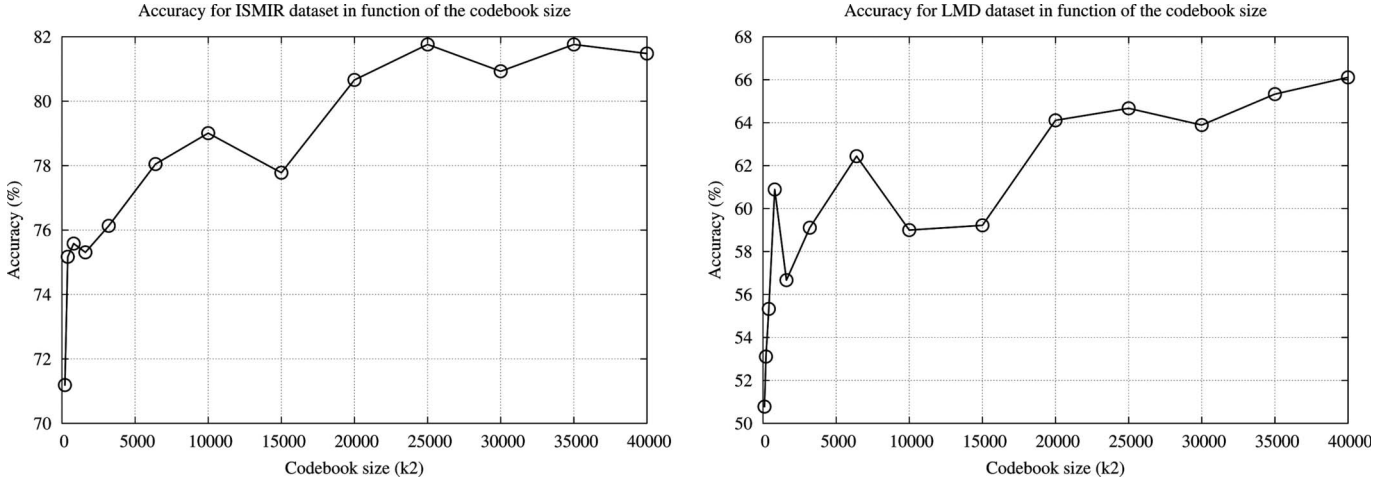

Fig. 1. Accuracy curves on the ISMIR test set (left) and on the LMD test set (right) obtained with codebooks of different sizes (x-axis), based on Sobol sequences. For the LMD dataset, the results are averaged over three groups.

Table 2. Results for the ISMIR04 dataset. Results obtained with codebooks generated with data from a single genre, with RR selection method. For comparison, the first line contains the results obtained with codebooks computed with all the genres.

ISMIR04 — only one genre			
	Markov	SVM	k-NN
all genres	83.03 ± 0.64	74.81 ± 1.29	74.72 ± 1.24
only-Class.	79.36 ± 0.72	71.40 ± 1.15	71.59 ± 1.28
only-Elec.	82.63 ± 0.73	73.54 ± 1.25	73.59 ± 0.65
only-JaBl	80.70 ± 0.53	73.29 ± 0.68	71.92 ± 0.67
only-MePu.	79.45 ± 0.85	71.22 ± 1.21	69.36 ± 0.80
only-RoPo.	81.88 ± 0.27	72.91 ± 0.97	72.19 ± 1.28
only-Wor.	81.26 ± 0.75	73.17 ± 1.50	73.20 ± 0.98

Table 3. Results for the LMD dataset. Results obtained with codebooks generated with data from a single genre, with RR selection method. For comparison, the first line contains the results obtained with codebooks computed with all the genres.

LMD — only one genre			
	Markov	SVM	k-NN
all genres	71.58 ± 1.41	62.63 ± 0.87	58.03 ± 2.92
only-Axe	71.84 ± 1.79	61.14 ± 1.04	58.06 ± 2.49
only-Bach.	70.68 ± 1.39	60.74 ± 1.10	55.76 ± 2.41
only-Bole.	68.92 ± 1.67	58.03 ± 0.96	53.52 ± 3.19
only-Forr.	71.61 ± 1.33	62.14 ± 1.06	57.91 ± 2.46
only-Gáuc.	71.99 ± 1.29	61.52 ± 0.89	57.56 ± 2.89
only-Mere.	70.72 ± 1.51	60.74 ± 0.95	55.98 ± 3.16
only-Pago.	71.89 ± 1.88	61.71 ± 1.28	56.60 ± 3.08
only-Sals.	71.02 ± 1.39	61.14 ± 0.78	56.84 ± 2.87
only-Seta.	71.58 ± 1.54	61.50 ± 1.20	57.81 ± 2.48
only-Tang.	53.81 ± 4.40	44.21 ± 2.76	38.48 ± 4.26

apparently no clear advantage in building class-dependent statistical models of frames. The last aspect is further explored in the next section.

4. Information-theoretic analysis

4.1 Statistical distributions of feature vectors

One of the findings of the previous section is that performance of classifiers is not altered even when the training instances in the codebook generation process are restricted to only one class. This raises questions on how the short-term feature vectors are distributed in the data representation space. Is there a strong class overlap? Are there genre-specific regions of the feature space? The objective of this section is to shed a light on these issues, through the analysis of the symbol distributions from an information-theoretic point of view. The entropy of the codebook gives us a measure of how the symbols are distributed among the clusters. The (normalized) symbol entropy is:

$$\mathcal{H}(S) = \frac{-1}{\log k_2} \sum_{k=1}^{k_2} P(s_k) \log(P(s_k)), \qquad (3)$$

where $P(s_k)$ is the a priori probability of symbol s_k (cluster s_k, with $k = 1, \ldots, k_2$). The normalizing constant $log k_2$, limits the entropy, $\mathcal{H}(S)$, to values in the interval [0, 1]. High entropy values are associated with codebooks that have even symbol distribution, while low entropy values correspond to codebooks with a few predominant clusters. The same reasoning can be applied to the conditional symbol entropy for a given genre. The (normalized) symbol entropies for a particular genre is:

$$\mathcal{H}(S|G_i) = \frac{-1}{\log k_2} \sum_{k=1}^{k_2} P(s_k|G_i) \log(P(s_k|G_i)), \qquad (4)$$

where $P(s_k|G_i)$ is the conditional probability of symbol k, given the genre G_i. In this case, high conditional entropy values are associated with a wide coverage of the data space by the feature vectors belonging to genre G_i. On the other hand, low values mean that the short-term instances of the genre are quantized by a restricted set of clusters, and therefore are confined to specific regions of the data space. This is confirmed by the results in Figure 2. Figure 2 shows the symbol distributions for the different classes in the ISMIR04 dataset, for codebooks created with frames from only one genre. Each line pertains to a single codebook, and in it the symbol distributions are represented by genre (columns). For comparison, in the last line are the genre distributions for a codebook created using all genres. The conditional entropy of the symbols is an indicator of the frame diversity of the genre. For instance, for the codebook based on Electronic music, the class distributions have a more uniform behaviour than the class distributions for the codebook based on Metal-Punk, which tells us that the frames in this genre are indeed less diversified. Figure 2 shows that the plots in the diagonal have the highest entropy in each line. This is natural since the codebooks were generated with frames belonging to the same genre, and therefore, the symbols for that specific genre are more evenly distributed among the codebook clusters. The figure also shows that there is a large number of clusters with a strong genre overlap. Similar behaviours were observed inspecting the codebooks for the LMD dataset. We conducted further tests to study the discriminative capacity of the feature vectors and to analyse to what extent they characterize musical genres. An intuitive measure of the clusters' discriminative capacity is the

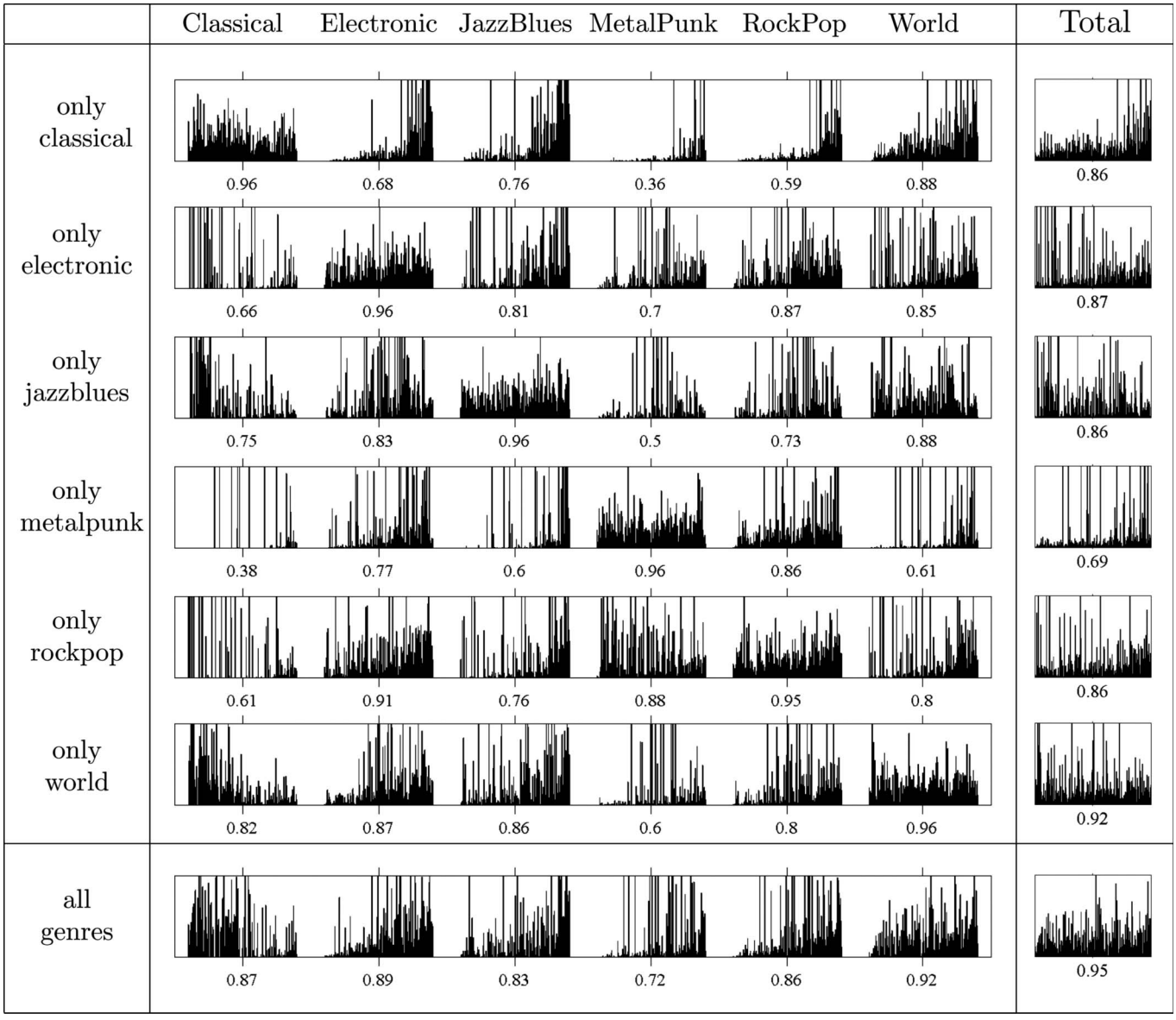

Fig. 2. Symbol distributions by genre for the ISMIR04 dataset, for several codebooks with $k_2 = 200$, created using frames from only one genre. For visualization purposes, y-axis on all the plots is limited to 0.02. The numbers beneath each plot are the conditional symbol entropies (Equation 4). In each line, the symbols were ordered in decreasing mutual information (Equation 5). The ordering is the same for all the plots in a given row.

mutual information, $\mathcal{I}(G; s_k)$, shared between a given symbol s_k and the genres:

$$\begin{aligned}\mathcal{I}(G; s_k) &= \mathcal{H}(G) - \mathcal{H}(G|s_k) \\ &= \mathcal{H}(G) + \sum_{i=1}^{|G|} P(G_i|s_k) \log(P(G_i|s_k)),\end{aligned} \tag{5}$$

where $|G|$ represents the total number of genres. $\mathcal{H}(G)$ is the entropy of the genres for all clusters, and is a fixed value greater or equal to the conditional entropy $\mathcal{H}(G|s_k)$. A high level of shared mutual information implies low values for the conditional entropy $\mathcal{H}(G|s_k)$. Low entropy values mean that the genre distribution for s_k is dominated by one of the genres. On the other hand, symbols with high entropy values have a weak discriminative capacity, since, in these clusters the genres tend to be approximately equiprobable.

Figure 3 represents the genre distribution given the symbols, for the LMD dataset, and for a codebook created with frames from all classes. The symbols were ordered by decreasing mutual information. High values correspond to dark shades, while low values are light-shaded. The figure shows that a majority of symbols exhibit a strong class overlap, and therefore have a low discriminative capacity. However a minority of symbols are class-dependent, with about half of these belonging to the genre Tango and the rest distributed among the remaining classes. The imbalance between Tango and the

other genres is due to the poor sound quality of the Tango tracks (recordings prior to 1935), and since almost no other pieces in the dataset were recorded that early on, this is sufficient to identify the genre. In this case, the classification is dependent on the recording quality rather than audio characteristics related to genre.

From these experiments, it remains unclear what contribution, if any, have the more discriminative symbols and their overall percentage in the performance of classification algorithms. These questions are addressed in the next section.

4.2 Discriminative codebook generation

The previous experiments showed that there are at least some minor dependencies between short-time frames and genres, since a small subset of symbols have predominant classes (although the vast majority appear in every class). In this section, we test the influence of genre information present in the symbols (or the lack thereof) on the overall classifier performances. Our objective is to ascertain if codebooks comprised of clusters with high discriminative capacity yield better classification accuracies than codebooks where the genres are evenly distributed among all clusters. To build such codebooks, we opted to first create a codebook with $k_2 = 400$, using frames from all genres, and selected a subset of the 200 cluster to generate a new codebook. The selection process was based on the mutual information between the clusters and the genres (Equation 5). In our experiments, three codebooks with $k_2 = 200$ were generated from a single codebook with $k_2 = 400$, by selecting the 200 most discriminative clusters (with the highest mutual information), the 200 less discriminative clusters, and the 200 in the middle. We are aware that quantizing the feature vectors with the new codebooks will change the symbol distribution, and, therefore, will change the mutual information contents in each cluster. Nevertheless, our observations suggest that codebooks created from a subset of clusters within a given discriminative range, will also have a high number of clusters within that range. For example, codebooks derived from the most discriminative clusters have, on average, 20% to 30% of clusters with mutual information values above 0.5 (on a scale of 0 to 1), while the least discriminative codebooks have about 5% (see Figure 4). Table 4 shows the accuracies obtained on the ISMR04

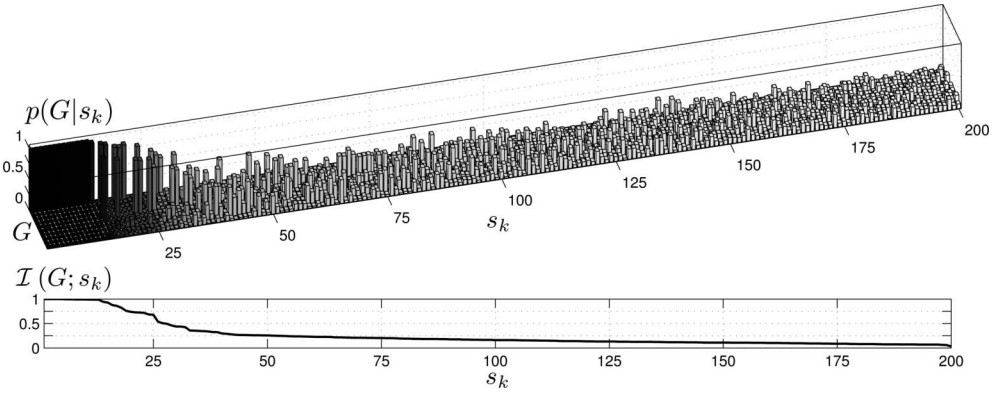

Fig. 3. Conditional class distribution, $p(G|s_k)$, for the LMD dataset for a codebook trained with all genres. The symbols were ordered by decreasing mutual information, $\mathcal{I}(G;s_k)$ (Equation 5), as show in the bottom plot. High mutual information values are dark-shaded, and low values light-shaded. Inspecting the leftmost symbols (e.g. the first 30 symbols have mutual information values above 0.44), around 15 symbols belong to the class Tango (the first set of symbols—last row), and the remaining symbols are distributed among the other classes.

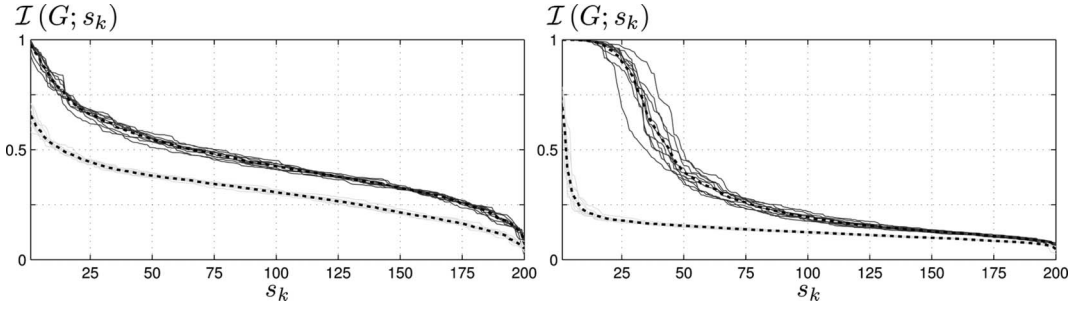

Fig. 4. Mutual information curves for codebooks created with a subset of 200 clusters from larger codebooks with $k_2 = 400$. In black are the curves of the most discriminative codebooks, and in grey the less discriminative, for ten test runs for the ISMIR04 dataset (left), and the LMD dataset (right). The dotted lines represent the mean values.

dataset, with the proposed codebook-generation method. The results show that there is not much variation in the classifiers' performances for the different codebooks. Similar behaviour was observed for the LMD dataset. Also it should be noted that the least discriminant codebooks for both datasets, have a significantly smaller number of discriminant symbols than their most discriminative counterparts, and still obtain similar performance values. This indicates that in codebook representations, each symbol discriminative capacity is not essential for genre classification.

5. Hubness

Hubness is a phenomenon that has been noticed in Music Similarity tasks and other pattern recognition applications such as fingerprint recognition (Aucouturier, 2006): some database samples, called 'hubs' appear in the neighbourhood of many of the database patterns leading to a large number of false positives. Flexer, Schnitzer, Gasser, and Pohle (2010) showed that the hub phenomenon was also present in large databases.

In this section we explore the influence of three parameterizations of the feature space on the hubness. In the first case we followed the same procedure described in Flexer et al. (2010) building a single Gaussian model for each music piece and used the symmetrized Kulback–Liebler divergence as a similarity measure (GMM + KL measure). In the second case, we generated a codebook following the RR procedure, calculated symbol histograms (Section 2.2) and used the Euclidean distance as a similarity measure (RR + Euclid measure) and the third one uses a codebook generated based on a Sobol sequence (SS + Euclid measure). Since our objective is not classification, both training and test samples were used.

Several criteria can be used to evaluate hubness. The *n*-hubness of a music piece (Flexer et al., 2010) is the number of music pieces *n*-neighbourhoods in which it appears. The *maxhub* is the largest *n*-hubness, i.e. the maximum number of *n*-neighbourhoods a single music piece belongs to. Figure 5 shows the *maxhub* for neighbourhoods ranging from 1 to 20, for both similarity measures and both ISMIR04 and LMD datasets.

Another criterion is the *hub3 %* which correspond to the percentage of the music pieces with *n*-

Table 4. Results for the ISMIR04 dataset, obtained with the discriminative codebooks (see Section 4.2). For comparison, the first line are the results presented in Table 1.

ISMIR — discriminative codebooks		
	Markov	SVM
all symbols	83.03 ± 0.64	74.81 ± 1.29
Top 200	82.54 ± 0.75	74.55 ± 1.09
Middle 200	83.05 ± 0.67	74.13 ± 1.08
Bottom 200	82.00 ± 0.70	74.02 ± 0.75

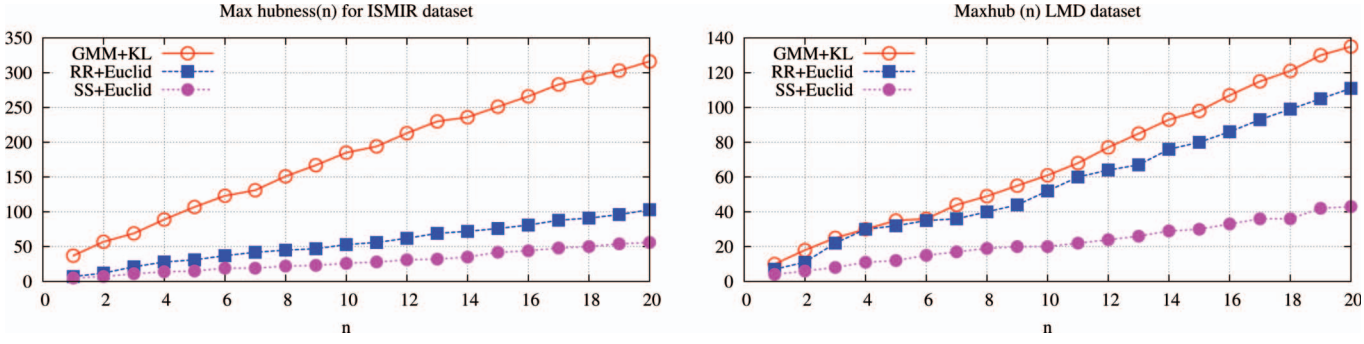

Fig. 5. The value of *maxhub* for neighbourhood sizes between 1 and 20 for three similarity measures.

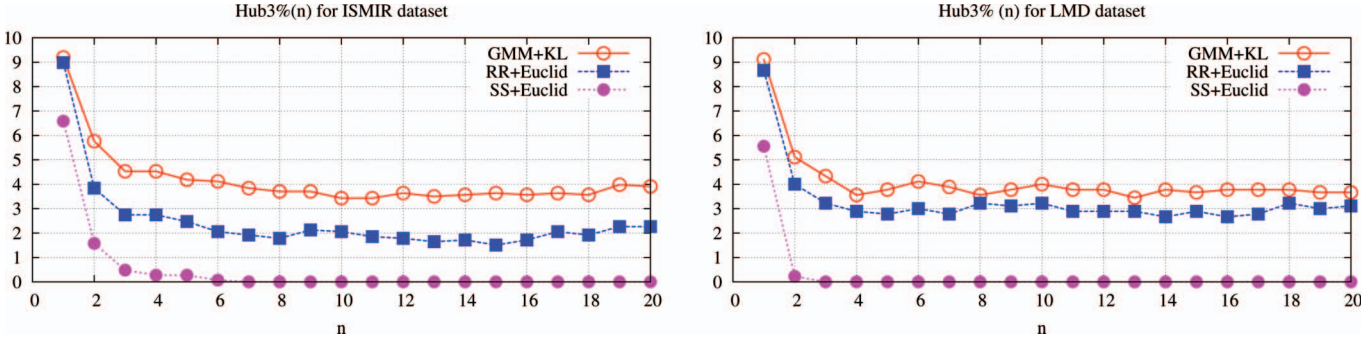

Fig. 6. The value of *hub3 %* as a function of the neighbourhood size for three similarity measures.

hubness larger than three times n. Figure 6 shows the *hub3 %* for both similarity measures and both datasets.

The *maxhub* can be viewed as an estimate of the 'worst case', for $n = 8$ the *maxhub* is equal to 150 for the GMM + KL similarity measure and less than 50 for the RR + Euclid similarity measure (ISMIR04). It can be seen that for both datasets, around 4% of music pieces create hubs when the GMM + KL similarity measure is used, while this number is around 2% (ISMIR04) and 3% (LMD) for RR + Euclid similarity measure. In all cases the SS + Euclid similarity measure generates much less hubs than the other two methods but the accuracy also tends to be lower (for the same codebook size). As was shown in Section 3, codebooks generated with the RR procedure perform at least as well as other alternatives in terms of accuracy, and one can see that it performs also better in terms of hubness.

6. Conclusions and future work

In this paper, we tackled the problem of music genre classification with low-level features computed on collections of audio frames. In the usual approach to this problem, it is implicitly assumed that specific regions of the data representation space defined by these features are representative of genres. Consequently, the process of partitioning the representation space—usually considered jointly with the design and training of genre models—appears instrumental to the performance of genre classifiers.

This paper precisely aims at evaluating the influence of diverse partitions of the representation space, given a fixed set of low-level audio features, on the performance of genre classifiers. We used a codebook approach to separate the task of partitioning the data representation space from that of training genre models, for which we tried different classifiers. The diverse partitions of the representation space (i.e. the diverse codebooks designed) used in our experiments were obtained through systematically varying the amount of information regarding the provenance of audio frames.

The main conclusion of these experiments is that there is no apparent benefit in seeking a thorough representation of genres in the low-level features representation space. A randomized and uninformed data representation permits one to build genre models that are as good as those built from thoroughly informed data representations. In addition to being simpler and requiring less computational resources to design, we showed that the former also has the advantage of producing less hubs than the latter.

These experiments and the knowledge gained from their analysis also led us to question the utility of separating the processes of partitioning the representation space and of training genre models. The codebook approach used for the task proved effective enough. Furthermore, the analysis of the symbols' distribution from information theoretic perspective showed that a majority of symbols have a strong class overlap, while only a few have predominant classes, which corroborates the premise that the feature space is dominated by regions that are common to all genres, and that genre specific areas are sparse. In addition, the symbols' discriminative capacity does not appear to be essential for genre classification, and hence the discriminative power of genre specific regions is also questionable.

It is also shown in this paper that taking into account the dynamics of the signal (via Markov-based classifiers) performs better than considering frames independently of their temporal ordering (via Support Vector Machines or k-NN working on frame histograms) (Meng, Ahrendt, Larsen, & Hansen, 2007; Langlois & Marques, 2009; Marques, Lopes, Sordo, Langlois, & Gouyon, 2010).

We believe that the results detailed in this paper contribute to the emerging idea that improvements in music genre classification will require the design of better initial signal representations, features that carry information that would be specific to genres closer to musical concepts (Aucouturier, 2009).

Acknowledgements

This research was supported by Convénio FCT/CAPES 2009; *Fundação para a Ciência e a Tecnologia* (FCT) and QREN-AdI grant for the project Palco3.0/3121 in Portugal; *Ministerio de Educación* in Spain. This work was partially supported by FCT through LASIGE Multiannual Funding and VIRUS research project (PTDC/EIA-EIA/101012/2008). The first author is supported by PROTEC grant SFRH/PROTEC/50118/2009.

References

Aucouturier, J.-J. (2006). *Dix expériences sur la modélisation du timbre polyphonique* (PhD thesis). University Paris VI, France.

Aucouturier, J.-J. (2009). Sounds like teen spirit: Computational insights into the grounding of everyday musical terms. In J. Minett & W. Wang (Eds.), *Language, Evolution and the Brain, Frontiers in Linguistics Series* (pp. 35–64). Taipei: Academia Sinica Press.

Aucouturier, J.-J., & Pachet, F. (2004). Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, *1*(1).

Aucouturier, J.-J., & Pachet, F. (2007). The influence of polyphony on the dynamical modelling of musical timbre. *Pattern Recognition Letters*, *28*(5), 654–661.

Berenzweig, A., Logan, B., Ellis, D., & Whitman, B. (2004). A large-scale evaluation of acoustic and subjective music similarity measures. *Computer Music Journal*, *28*(2), 63–76.

Cano, P., Gomez, E., Gouyon, F., Herrera, P., Koppenberger, M., Ong, B., Serra, X. Streich, S., & Wack, N. (2006). ISMIR 2004 audio description contest. In *MTG Technical Report MTG-TR-2006-02*, Music Technology Group, Universitat Pompeu Fabra, Spain.

Chen, K., Gao, S., Zhu, Y., & Sun, Q. (2006). Music genres classification using text categorization method. In *2006 IEEE 8th Workshop on Multimedia Signal Processing (MMSP)*, Victoria, Canada, pp. 221–224.

El-Manzalawy, Y., & Honavar, V. (2005). *WLSVM: Integrating LibSVM into Weka Environment*. Software retrieved from http://www.cs.iastate.edu/~yasser/wlsvm

Flexer, A. (2007). A closer look on artist filters for musical genre classification. In *ISMIR 2007 – 8th International Conference on Music Information Retrieval*, Vienna, Austria, pp. 341–344.

Flexer, A., Schnitzer, D., Gasser, M., & Pohle, T. (2010). Combining features reduces hubness in audio similarity. In *ISMIR 2010 – 11th International Conference on Music Information Retrieval*, Utrecht, the Netherlands, pp. 171–176.

Hoffman, M., Blei, D., & Cook, P. (2009). Easy as CBA: A simple probabilistic model for tagging music. In *ISMIR 2009 – 10th International Conference on Music Information Retrieval*, Kobe, Japan, pp. 369–374.

Langlois, T., & Marques, G. (2009). Music classification method based on timbral features. In *ISMIR 2009 – 10th International Conference on Music Information Retrieval*, Kobe, Japan, pp. 81–86.

Lidy, T., & Rauber, A. (2005). Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *ISMIR 2005 – 6th International Conference on Music Information Retrieval*, London, UK, pp. 34–41.

Logan, B., & Salomon, A. (2001). A music similarity function based on signal analysis. In *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME 2001*, Tokyo, Japan, pp. 190–193.

Lopes, M., Gouyon, F. Silla, C., & Oliveira, L.E.S. (2010). Selection of training instances for music genre classification. In *ICPR 2010: 20th International Conference on Pattern Recognition*, Istanbul, Turkey, pp. 4569–4572.

Marques, G., Lopes, M., Sordo, M., Langlois, T., & Gouyon, F. (2010). Additional evidence that common low-level features of individual audio frames are not representative of music genre. In *SMC 2010 – 7th Sound and Music Computing Conference*, Barcelona, Spain, pp. 134–139.

Meng, A., Ahrendt, P., Larsen, J., & Hansen, L.K. (2007). Temporal feature integration for music genre classification. *IEEE Transactions on Audio, Speech, and Language Processing*, *15*(5), 1654–1664.

Pampalk, E. (2006). *Computational models of music similarity and their application in music information retrieval* (PhD thesis). Vienna University of Technology, Austria.

Pampalk, E. Flexer, A., & Widmer, G. (2005). Improvements of audio-based music similarity and genre classification. In *ISMIR 2005 – 6th International Conference on Music Information Retrieval*, London, UK, pp. 628–633.

Panagakis, Y., & Arce, G. (2009). Music genre classification using locality preserving non-negative tensor factorization and sparse representations. In *ISMIR 2009 – 10th International Conference on Music Information Retrieval*, Kobe, Japan, pp. 249–254.

Seyerlehner, K., Widmer, G., & Knees, P. (2008). Frame level audio similarity – a codebook approach. In *11th International Conference on Digital Audio Effects DAFx 2008*, Espoo, Finland, pp. 349–356.

Silla, C., Koerich, A., & Kaestner, C. (2008). The latin music database. In *ISMIR 2008 – 9th International Conference on Music Information Retrieval*, Philadelphia, PA, USA, pp. 451–456.

Soltau, H., Schultz, T., Westphal, M., & Waibel, A. (1998). Recognition of music types. In *ICASSP'98 IEEE International Conference on Acoustics, Speech and Signal Processing*, Seattle, WA, USA, Vol. II, pp. 1137–1140.

Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio singals. *IEEE Transactions on Speech and Audio Processing*, *10*(5), 293–302.

West, K., & Cox, S. (2005). Finding an optimal segmentation for audio genre classification. In *ISMIR 2005 – 6th International Conference on Music Information Retrieval*, London, UK, pp. 680–685.