ADDITIONAL EVIDENCE THAT COMMON LOW-LEVEL FEATURES OF INDIVIDUAL AUDIO FRAMES ARE NOT REPRESENTATIVE OF MUSIC GENRE

Gonçalo Marques[1], **Miguel Lopes**[2], **Mohamed Sordo**[3], **Thibault Langlois**[4], **Fabien Gouyon**[2]
[1]DEETC - ISEL Lisboa, [2]INESC Porto, [3]Universitat Pompeu Fabra, Barcelona, [4]DI - FCUL Lisboa
[1]`gmarques@isel.pt`

ABSTRACT

The Bag-of-Frames (BoF) approach has been widely used in music genre classification. In this approach, music genres are represented by statistical models of low-level features computed on short frames (e.g. in the tenth of ms) of audio signal. In the design of such models, a common procedure in BoF approaches is to represent each music genre by sets of instances (i.e. frame-based feature vectors) inferred from training data. The common underlying assumption is that the majority of such instances do capture somehow the (musical) specificities of each genre, and that obtaining good classification performance is a matter of size of the training dataset, and fine-tuning feature extraction and learning algorithm parameters.

We report on extensive tests on two music databases that contradict this assumption. We show that there is little or no benefit in seeking a thorough representation of the feature vectors *for each class*. In particular, we show that genre classification performances are similar when representing music pieces from a number of different genres with the same set of symbols derived from a single genre or from all the genres. We conclude that our experiments provide additional evidence to the hypothesis that common low-level features of isolated audio frames are not representative of music genres.

1. INTRODUCTION

A large literature exists on automatic classification of music pieces based on raw audio data. Providing a complete review is out of the scope of this paper, interested readers are referred to [1] and [2]. Most approaches to date share the same underlying algorithmic architecture [1]: the Bag-of-Frame (BoF) approach. Music genres are represented via long-term statistical models of large collections of feature vectors computed on short frames of audio signal (on the scale of tenth of ms). In the BoF approach, it is implicit that all frames have a similar information load, and that all are significant in the modeling of genres. A prototypical implementation of this architecture, now considered standard procedure, uses Gaussian Mixture Modeling (GMMs) of short-term Mel-Frequency Cepstral Coefficients (MFCCs).

Aucouturier [1] discusses the existence of a “glass-ceiling” to the performance of the BoF approach to music genre classification. He argues that it is fundamentally bound and that a change of paradigm to music genre modeling is needed. A number of recent papers also challenge the BoF approach arguing that all frames may not have the same information load and propose to train models of music genre on a selection of the available training data, either the most representative, or the most discriminative [3, 4, 5]. In previous research on the topic of instance selection for music genre classification [6], we showed that when representing music signals by common low-level features (i.e. MFCCs and spectral features), similar classification accuracies could be obtained when training classifiers on all of the training data available, or on few training data instances from each class.

In this paper, we go a step further and propose the hypothesis that values of common low-level features on isolated signal frames are *not* representative of music genres. In other words, the performance of BoF music genre models may be bound because there would be not such thing as a “typical e.g. Rock, or Classical frame” (more precisely, no such thing as “typical sets of low-level audio feature values for a e.g. Rock, or Classical frame”).

To address this hypothesis, we conduct systematic experiments in which models of music genres are built with training data representative of *only part of the genres* from the dataset.

These experiments imply (1) the definition of a codebook, generated from different partitions of some available training data (section 2), and (2) the expression of training examples from each genre with this codebook and the actual training of genre models (section 3.1). In the remainder of section 3, we describe experimental details regarding data and audio features used. Section 4 reports results of our experiments and in section 5, we propose some conclusions and directions for future research.

2. CODEBOOK GENERATION PROCEDURES

Following the technique described in [5], the experiments reported in this paper are based on a codebook approach. The feature vectors extracted from the training set are processed in order to select a limited number of representative feature vectors that constitute the codebook. We experimented several approaches for the constitution of the codebook, including selecting the centroids obtained with k-means, selecting the most representative feature vectors according to the probability density function modeled with a Gaussian Mixture Model, and combinations of both approaches (see [5] for more details). In this paper, to avoid any particular bias, we use random selection of feature vectors, as follows.

Codebooks are generated by randomly selecting k_1 feature vectors from each music piece and then selecting k_2 feature vectors in the set of $N \times k_1$ feature vectors (where N corresponds to the number of music pieces in the training set). In both cases a uniform distribution is used. For all experiments described in this paper we used $k_1 = 20$ and $k_2 = 200$.

Notice that the creation of codebooks is an unsupervised process, i.e. each music piece is processed independently of the class it belongs to. Three kinds of codebooks were generated:

Using data from all genres but one The codebook is generated ignoring class X. This is repeated for each class. The codebooks obtained this way are called “all-but-X”.

Using data from a single genre In this case codebooks are generated using the feature vectors found in music pieces from only one genre. These codebooks are referred as “only-X”.

Using data from all genres As a base for comparison, we generated codebooks that use the data from all classes, as described previously. These codebooks are called “all-genres”.

3. EXPERIMENTS

3.1 Classification models

3.1.1 Data representation by vector quantization

Input data to the classifiers (both for training and testing) is based on a codebook approach: each music piece is first converted into a sequence of discrete symbols pertaining to one of the codebook symbols considered here, through vector quantization of the audio features. More precisely, for each music piece, the feature vector of each frame is assigned a symbol corresponding to the nearest symbol in the set of $k_2 = 200$ possible symbols of the given codebook (see section 2).

Finally, depending on the classifier, each music piece is represented by either a normalized histogram of the symbols frequency, or the sequence of symbols itself.

3.1.2 Histogram + k-NN

The k-NN algorithm treats the histograms as points in a k_2 dimensional space. The music pieces in the training set are used as examples, and a new music piece is classified by a majority vote of its neighbors. In our experiments, we used a 5-NN classifier. The nearest neighbors were calculated based on two distance metrics: the Euclidean distance, and a symmetric version of the Kullback-Leibler divergence:

$$\mathcal{D}\left(P\|Q\right) = D_{KL}\left(P\|Q\right) + D_{KL}\left(Q\|P\right) \tag{1}$$

where, Q and P are the normalized histograms of two music pieces, and

$$D_{KL}\left(P\|Q\right) = \sum_{i=1}^{k_2} P(i) \log \frac{P(i)}{Q(i)} \tag{2}$$

is the Kullback-Leibler divergence, and $P(i)$ is the i-bin of the histogram P. In order to use this divergence, the distributions P and Q must have non-zero entries. However, this can happen if one or more symbols from the codebook are not used in the representation of a given music piece. To overcome this limitation, we add one hit to all histogram bins before performing the histogram normalization.

3.1.3 Histogram + SVM

A Support Vector Machine (SVM) [7] 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 $C = 2000$. Experiments for determining optimal parameter values are left for future work.

3.1.4 Markov models

This classification method is based on Markov modeling. A Markov model is build for each genre, by computing the transition probabilities (bigrams) for each group of sequences [5]. The outcome is set of transition matrices, one for each class, containing the probabilities, $P(s_j|s_i)$, of each symbol s_j given the preceding symbol s_i. For classification, the test music piece is converted into a sequence of symbols, $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, and the (logarithmic) probability of the sequence given each model is calculated:

$$\begin{aligned}\mathcal{L}_M(\mathcal{S}) &= \log\left(P_M(s_{i=1,\ldots,n})\right)\\ &= \log\left(P_M(s_1)\right) + \sum_{i=2}^{n} \log\left(P_M(s_i|s_{i-1})\right)\end{aligned} \tag{3}$$

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 $\mathcal{L}_M$.

3.2 Data

Two datasets were used in our experiments. 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”).

3.2.1 LMD

The Latin Music Database [8] is composed of 3,227 full-length music pieces, uniformly distributed over 10 genres: Axé, Bachata, Bolero, Forró, Gaúcha, Merengue, Pagode, Salsa, Sertaneja, and Tango. For our experiments, we created a subset of 900 music pieces, which are divided in three folds of equal size (30 pieces per class). We used an artist filter [9, 10] to ensure that the music pieces from a specific artist are present in one and only one of the three folds. We also added the constraint of the same number of artists per fold.

3.2.2 ISMIR04

This dataset was created for the genre classification contest organized during the ISMIR 2004 conference [11], [1] and is divided in six genres with a total of 729 music pieces for training and 729 music pieces for testing. The music piece distribution among the six genres is: 320 Classical, 115 Electronic, 26 JazzBlues, 45 MetalPunk, 101 RockPop, and 122 World. As in the original ISMIR 2004 contest, the dataset does not account for artist filtering between both sets.

3.3 Audio Features

We used the free MARSYAS framework [2] to extract 17 audio features from 46ms frames of the audio signals (mono, sampled at 22050Hz, no overlap). The features are commonly used in audio genre classification tasks: the zero crossing rate, spectral centroid, rolloff frequency, spectral flux, and 13 MFCCs, including MFCC0.

3.4 Evaluation Metrics

We report the accuracy obtained over test sets only, both for the ISMIR04 and LMD datasets.

On the evaluation on the ISMIR04 dataset, we kept the original training-testing division proposed in the ISMIR 2004 genre classification contest.

The evaluation on the LMD dataset follows a three-fold cross validation procedure: two folds are used for training and one for testing, with all the permutations of the folds. The performance measure is the accuracy averaged over the three test runs.

4. RESULTS AND DISCUSSION

First of all, it is interesting to notice that results obtained on the ISMIR04 and LMD datasets are comparable to state-of-the-art results. For instance, the best result obtained on ISMIR04 is 79.8%, which is very similar to the results obtained by the best algorithms in the last MIREX in which this data was used (i.e. MIREX 2005). [3] The best result obtained on LMD is 64.9%, when the best result on this dataset in MIREX 2009 was 74.6% and the average accuracy accounting for all participants was 55.5%.

In almost every set of experiment we found that the classifiers based on Markov models is better than the three other alternatives. This observation tends to confirm the fact that the information contained in the temporal sequence is indeed relevant to the classification into genres.

Tables 1 and 2 show the overall classification accuracy for the ISMIR04 and the LMD datasets respectively when one genre is not used in the codebook generation process. The lines represent accuracy scores obtained with different classification procedures (columns). For the sake of comparison, the first line contains the results obtained with a codebook computed with all genres.

ISMIR04 — all-but-one genre				
	Markov	SVM	k-NN	k-NN$_{\mathrm{KL}}$
all genres	79.0	**68.6**	73.0	**76.0**
all-but-Class.	**79.3**	68.3	70.8	73.7
all-but-Elec.	**79.7**	**68.6**	69.3	73.7
all-but-JaBl.	78.3	67.8	70.6	74.6
all-but-MePu.	**79.3**	68.5	71.5	75.9
all-but-RoPo.	78.6	68.2	**73.3**	75.3
all-but-Wor.	**79.1**	68.2	**73.3**	74.3
Average	79.1	68.3	71.5	74.6

Table 1. Results for the ISMIR04 dataset. Each line represents the average accuracy (over all genres) obtained with codebooks generated from all but one genre. The last line contains the average of lines 2 to 7. The first line contains the results obtained with a codebook computed with all genres. Results in bold are those that outperform those of the first line.

From these experiments (Tables 1 and 2) one can see that when the feature vectors from one class are ignored in the creation of the codebook, we do not observe a severe decrease of the accuracy. In some cases the accuracy obtained without one of the classes is equal or better than when all genres are used (numbers with bold font).

LMD — all-but-one genre				
	Markov	SVM	k-NN	k-NN$_{\mathrm{KL}}$
all genres	64.0	54.2	47.0	52.2
all-but-Axé	62.7	**56.6**	**50.1**	**52.7**
all-but-Bach.	**64.9**	54.1	**48.1**	**52.7**
all-but-Bole.	63.0	**54.9**	**49.9**	**53.0**
all-but-Forr.	61.1	53.8	**48.9**	51.2
all-but-Gáu.	63.6	53.1	**48.3**	50.9
all-but-Mer.	63.7	53.6	**47.8**	50.4
all-but-Pag.	63.5	54.0	**49.1**	**52.7**
all-but-Sals.	63.5	53.7	**47.9**	50.8
all-but-Sert.	62.8	**54.3**	**48.9**	51.4
all-but-Tan.	63.1	**54.3**	**48.9**	51.9
Average	63.2	54.2	48.8	51.8

Table 2. Results for the LMD dataset. Each line represents the average accuracy (over all genres) obtained with codebooks generated from all but one genre. The last line contains the average of lines 2 to 11. The first line contains the results obtained with a codebook computed with all genres.

Tables 3 and 4 are very similar to table 1 and 2, but in this case, the codebooks were computed using feature vectors from only one genre.

It can be seen that reducing dramatically the universe of feature vectors, the average accuracy compared to the case where all genres are used is not substantially different.

[1] http://ismir2004.ismir.net/ISMIR_Contest.html
[2] http://marsyas.sness.net/
[3] http://www.music-ir.org/mirex/2005/

ISMIR04 — only one genre				
	Markov	SVM	k-NN	k-NN$_{\mathrm{KL}}$
all genres	79.0	**68.6**	**73.0**	**76.0**
only-Class.	76.3	62.8	66.8	71.7
only-Elec.	**79.4**	67.2	70.9	73.4
only-JaBl	78.5	66.8	67.2	72.7
only-MePu.	74.8	66.9	65.3	72.3
only-RoPo.	75.7	67.2	70.5	75.4
only-Wor.	**79.8**	67.5	69.8	73.3
Average	77.4	66.4	68.4	73.1
All − Average	1.6	2.2	4.6	2.9

Table 3. Results for the ISMIR04 dataset. Each line represents the accuracy obtained with codebooks generated with data from a single genre. The last line shows the decrease in accuracy between results obtained with a codebook generated with data from all genres and average results obtained with a codebook generated with data from a single genre.

LMD — only one genre				
	Markov	SVM	k-NN	k-NN$_{\mathrm{KL}}$
all	64.0	54.2	47.0	52.2
only-Axé	59.9	**54.4**	**49.6**	**52.2**
only-Bach.	**64.9**	53.7	45.8	51.7
only-Bole.	62.1	53.0	45.6	49.0
only-Forr.	**64.7**	53.9	**49.6**	**52.9**
only-Gáu.	**64.6**	**54.6**	**49.8**	**55.8**
only-Mer.	62.3	53.6	**48.7**	51.0
only-Pag.	63.5	53.0	**48.5**	50.8
only-Sals.	63.9	53.1	**47.2**	50.9
only-Sert.	61.9	53.3	**49.2**	**54.3**
only-Tan.	55.0	46.2	40.1	43.6
Average	62.3	52.9	47.4	51.2

Table 4. Results for the LMD dataset. Each line represents average results obtained with codebooks generated with data from a single genre.

In the case of the ISMIR04 dataset, using only one genre for building the codebook leads to an average decrease of 1.6 percentage points for Markov models, 2.2 percentage points for SVM, 4.6 percentage points for k-NN and 2.9 percentage points for k-NN$_{\mathrm{KL}}$. It is interesting to note that the non-parametric method (k-NN) is the most affected by a reduction of the amount of data. However, we can also see that, at least for the Markov model classifier, in some cases performance can be better when using only one genre to build the codebook.

In the case of the LMD dataset (Table 4), we observe that, in numerous cases, the accuracy obtained with codebooks modeled after only one genre is equal or better than the one obtained using all genres. From these experiments we can see that using a small subset of the feature vectors, even if they belong to only one genre, we are still able to build a classifier that performs well.

ISMIR04				
	all	all-but-1	only-1	Diff.
Classical	**96.9**	95.3	96.6	1.3
Electronic	71.1	72.8	**73.7**	0.9
JazzBlues	63.4	69.3	**76.9**	7.6
MetalPunk	71.1	**75.6**	64.4	−11.2
RockPop	61.8	61.8	**64.7**	2.9
World	59.9	60.7	**63.1**	2.4

Table 5. Breakdown with respect to genres of the results for the ISMIR04 dataset, using Markov models classifiers. Each row shows the accuracy observed for the corresponding class with the three different kinds of codebooks. For instance, the entry in the fourth line second column (75.6) is the percentage of correctly classified music pieces for the class MetalPunk, using a codebook computed with feature vectors from all genres but MetalPunk. The last column contains the difference between the only-1 and all-but-1 accuracies.

LMD				
	all	all-but-1	only-1	Diff.
Axé	**44.4**	41.1	36.7	−4.4
Bach.	83.3	84.4	**86.7**	2.3
Bole.	76.7	74.4	**82.2**	7.8
Forr.	**37.8**	35.6	35.6	0.0
Gaúc.	44.4	47.8	**53.3**	5.5
Mere.	80.0	76.7	**81.1**	4.4
Pago.	56.7	56.7	**60.0**	3.3
Sals.	68.9	65.6	**72.2**	6.6
Sert.	61.1	54.4	**64.4**	10.0
Tang.	**86.7**	85.6	85.6	0.0

Table 6. Breakdown with respect to genres of the results for the LMD dataset, using Markov models classifiers. Each row show the accuracy observed for the corresponding class with the three different kinds of codebooks. The last column contains the difference between the only-1 and all-but-1 accuracies.

Since the performance is measured on all classes, a lower classification rate on one class may be hidden by higher scores on others. Therefore we evaluated the accuracy obtained for each class with each of the three ways of building the codebooks. These results are shown in tables 5 and 6. In the case of the ISMIR04 dataset (table 5), one can see that the differences in accuracy between using only a given class (4th column) and not using that class at all in the generation of the codebook (3rd column) is rather small with two exceptions: JazzBlues and MetalPunk, albeit in an opposite way. These exceptions may be explained by the fact that both categories are represented with a very small number of music pieces (26 for JazzBlues and 45 for MetalPunk).

We also studied the effect of using only one class for codebook creation on the accuracy observed on other classes. The results are shown in table 7 for the ISMIR04 dataset

ISMIR04							
	only-Class.	only-Elec.	only-JaBu	only-MePu.	only-RoPo.	only-Wor.	Diff.
Class.	**96.6**	95.9	95.6	91.3	91.6	95.9	0.7
Elec.	69.3	**73.7**	71.1	67.5	71.1	72.8	0.9
JaBu.	57.7	69.2	**76.9**	61.5	65.4	65.4	7.7
MePu.	77.8	73.3	77.8	64.4	73.3	**82.2**	4.4
RoPo.	51.0	61.8	59.8	63.7	**64.7**	59.8	1.0
Wor.	54.1	60.7	56.6	54.1	50.8	**63.1**	2.4

Table 7. Genre breakdown results for the ISMIR04 dataset using Markov models with different codebooks based on only one class. The table entries are class accuracies (lines) for a given codebook (columns). The last column shows the difference between the best (bold) and the second best accuracy (underlined) of each row.

LMD											
	only-Ax.	only-Ba.	only-Bo.	only-Fo.	only-Gá.	only-Me.	only-Pa.	only-Sa.	only-Se.	only-Ta.	Diff.
Axé	36.5	42.2	22.1	44.4	45.6	**46.7**	37.8	44.4	45.6	37.8	1.1
Bach.	81.1	**86.7**	83.3	83.3	84.4	83.3	85.6	84.4	81.1	83.2	1.1
Bole.	76.7	77.8	**82.2**	68.9	74.4	70.0	70.0	72.2	68.9	80.0	2.2
Forr.	34.4	34.4	32.2	35.6	32.2	34.4	**41.1**	33.3	32.2	36.7	4.4
Gáu.	44.4	51.1	43.3	**53.3**	**53.3**	48.9	44.4	51.1	45.6	33.3	2.2
Mer.	75.6	76.7	77.8	80.0	77.8	**81.1**	75.6	77.8	76.7	72.2	1.1
Pag.	50.0	57.8	55.6	**60.0**	**60.0**	48.9	**60 .0**	57.8	53.3	50.0	2.2
Sals.	61.1	**74.4**	72.2	72.2	72.2	63.3	61.1	40.3	65.6	42.2	2.2
Sert.	52.2	61.1	55.6	**64.4**	58.9	62.2	58.9	60.0	**64.4**	28.9	2.2
Tan	86.7	86.7	84.4	84.4	87.8	84.4	**88.9**	85.6	85.6	85.6	1.1

Table 8. Genre breakdown results for the LMD dataset using Markov models with different codebooks based on with only one class. The table entries are class accuracies (lines) for a given codebook (columns). The last column shows the difference between the best (bold) and the second best accuracy (underlined) of each row.

and in table 8 for the LMD dataset. Each row of these tables contain the accuracy observed on one class (rows) when using a codebook based on each single class (columns). Values in bold font correspond to the maximum of each row and can be interpreted as the best codebook for the representation of each class. For example, in table 7 we can see that all classes but MetalPunk are better represented by a codebook defined using the same class. But if we look at the second best accuracy (underlined numbers) we can see that using feature vectors from a different class can lead to seemingly similar performance. The difference between best and second best accuracy is shown in the last column. For example, Classical music may be represented by feature vectors that belong to Electronic or World music losing only 0.7 percentage points in accuracy. RockPop may be represented by MetalPunk or Electronic feature vectors, losing only 1 percentage point. Counter examples can be found with the cases of JazzBlues and MetalPunk but this may be caused by the fact that those classes are represented by a small number of music pieces when compared to other classes. It is notable that in some cases (such as Classical and Electronic) the ability of using a genre to represent another genre occurs with genres that are perceived very differently by listeners.

When looking at table 8, which describes the same experiments with the LMD dataset, one can see that the difference between the best and the second best accuracy is small, showing that a genre may be represented using feature vectors from another genre without losing too much accuracy, and in some cases even increasing accuracy apparently.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we tackle the problem of music genre classification with low-level features computed on collections of audio frames. In the common approach to this problem, it is generally assumed that the majority of frames of a particular genre (or, more precisely, their representations via MFCCs and other common low-level features) carry information that is specific to that genre. The main conclusion of our experiments is that common low-level features computed on individual audio frames are in fact not representative of music genres (even if their distributions are). We demonstrate that seeking the most extensive and thorough representation of each genre with respect to such low-level features does in fact not bring any benefit in terms of classification accuracy.

Specifically, in our experiments, music pieces from diverse genres are represented by sequences of symbols from a codebook. This codebook is made up of feature vectors from either one, all, or a selection of genres. We show that the classification accuracy averaged over all genres is very similar when the codebook is derived from data of all genres vs. data of all genres but one (tables 1 and 2), or vs. data of only one single genre (tables 3 and 4). This appears to be true for diverse classifiers. Further, the provenance of the data used for deriving the codebook does not seem to have a profound impact on classification accuracy of each particular genre (tables 5 and 6), even in the case where the data used comes from one single, different genre (tables 7 and 8). These results appear to hold for two different datasets of very different natures.

This is not to say that such frame-based representations are not useful for music genre classification, as they indeed permit to classify better than random. However, even if *collections* of frames can represent music genres with some success, we show here that *individual* frames do not.

Given the relatively small variations in accuracies for a given genre, and the fact that these variations go both ways (small decrease in some cases and small increase in some others), we suspect that statistical significance tests would show the near equivalence of accuracies over each genres. This is an avenue for future work.

We believe that the results detailed in this paper contribute to the emerging idea that significant 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 [12].

6. ACKNOWLEDGMENTS

Thanks to Jaime Cardoso at FEUP/INESC Porto, Alessandro L. Koerich at PPGIA-PUCPR and Luiz E. S. Oliveira at UFPR Curitiba, and to anonymous reviewers. 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.

7. REFERENCES

[1] J.-J. Aucouturier, *Dix expériences sur la modélisation du timbre polyphonique*. PhD thesis, University Paris VI, 2006.

[2] N. Scaringella, G. Zoia, and D. Mlynek, “Automatic genre classification of music content,” *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 133–141, 2006.

[3] J.-J. Aucouturier, B. Defreville, and F. Pachet, “The bag-of-frames approach to audio patter recognition: A sufficient model for urban soundscapes but not for polyphonic music,” *Journal of the Acoustical Society of America*, vol. 122, no. 2, pp. 881–891, 2007.

[4] U. Bagci and E. Erzin, “Automatic classification of musical genres using inter-genre similarity,” *IEEE Signal Processing Letters*, vol. 14, no. 8, pp. 521–524, 2007.

[5] T. Langlois and G. Marques, “Music classification method based on timbral features,” in *International Conference on Music Information Retrieval*, 2009.

[6] M. Lopes, F. Gouyon, C. Silla, and L. E. S. Oliveira, “Selection of training instances for music genre classification,” in *International Conference on Pattern Recognition*, 2010.

[7] Y. EL-Manzalawy and V. Honavar, *WLSVM: Integrating LibSVM into Weka Environment*, 2005. Software available at `http://www.cs.iastate.edu/~yasser/wlsvm`.

[8] C. Silla, A. Koerich, and C. Kaestner, “The latin music database,” in *International Conference on Music Information Retrieval*, 2008.

[9] E. Pampalk, *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, Austria, 2006.

[10] A. Flexer, “A closer look on artist filters for musical genre classification,” in *International Conference on Music Information Retrieval*, 2007.

[11] P. Cano, E. Gomez, F. Gouyon, P. Herrera, M. Koppenberger, B. Ong, X. Serra, S. Streich, and N. Wack, “Ismir 2004 audio description contest,” in *MTG Technical Report MTG-TR-2006-02*, 2006.

[12] J.-J. Aucouturier, “Sounds like teen spirit: Computational insights into the grounding of everyday musical terms,” in *Language, Evolution and the Brain, Frontiers in Linguistics Series* (J. Minett and W. Wang, eds.), Taipei: Academia Sinica Press, 2009.