

# A text categorization approach for music style recognition

**Abstract.** The automatic classification of music files into styles is one challenging problem in music information retrieval and for music style perception understanding. It has a number of applications, like the indexation and exploration of musical databases. Some techniques used in text classification can be applied to this problem. The key point is to establish a music equivalent to the words in texts. A number of works use the combination of intervals and duration ratios for music description. In this paper, different statistical text recognition algorithms are applied to style recognition using this kind of melody representation, exploring their performance for different word sizes and statistical models.

## 1 Introduction

Machine learning and pattern recognition techniques, successfully employed in other fields, can be also applied to music analysis. One of the tasks that can be posed is the modelization of the music style. Immediate applications are the classification, indexation, and content-based search in digital music libraries, where digitised (MP3), sequenced (MIDI) or structurally represented (XML) music can be found. Some recent papers explore the capabilities of these methods to recognise music style, either using audio [1,2,3], or symbolic sources [4,5,6].

Our aim is to explore the capabilities of text categorization algorithms to solve problems relevant to computer music. In this paper, some of those methods are applied to the recognition of musical genres from a symbolic representation of the melody. Some styles like jazz, classical, ragtime or gregorian have been chosen as an initial benchmark for the proposed methodology due to the general agreement in the musicology community about their definition and limits.

## 2 Methodology

### 2.1 Data sets

Experiments were performed using two different corpora. Both of them are made up of MIDI files, containing monophonic sequences.

The first corpus is a set of MIDI files from *Jazz* and *Classical* music collected from different web sources, without any processing before entering the system. The melodies are real-time sequenced by musicians, without quantization. The corpus is made up of 110 MIDI files, 45 of them being classical music and 65 being jazz music. The length of the corpus is around 10,000 bars (40,000 beats).

The second corpus is that used by Cruz et al. [4]. It consists of 300 MIDI files, from three different styles: *Gregorian*, passages from the sacred music of *J. S. Bach* (Baroque), and *Scott Joplin* ragtimes; with 100 files per class. In this corpus, the melodies are step-by-step sequenced, and much shorter (around 10 bars per file in average) than in the former case.

## 2.2 Encoding

The encoding used in this work has been inspired in the encoding method proposed in [7]. This method generates  $n$ -grams of notes from the melodies, encoding pitch interval information and a kind of duration ratios. The encoding of MIDI files is performed as follows:

*Melody extraction.* The melody track from each MIDI file is analyzed, obtaining the pitch and duration for each note. Durations are calculated as the number of ticks between the onset of the note and that of the next, ignoring all intermediate rests. As a result, a pair of values  $\{pitch, duration\}$  is obtained for each note in the analyzed track.

*$n$ -word extraction.* Next, the melody is divided into  $n$ -note windows. For each window, a sequence of intervals and duration ratios is obtained (see Fig. 1 for an example), calculated using Eqs. (1) and (2) respectively.

$$I_i = Pitch_{i+1} - Pitch_i \quad (i = 1, \dots, n-1) \quad (1)$$

$$R_i = \frac{Onset_{i+2} - Onset_{i+1}}{Onset_{i+1} - Onset_i} \quad (i = 1, \dots, n-1) \quad (2)$$

Each  $n$ -word is defined then as a sequence of symbols:  $[I_1 R_1 \dots I_{n-1} R_{n-1}]$ .

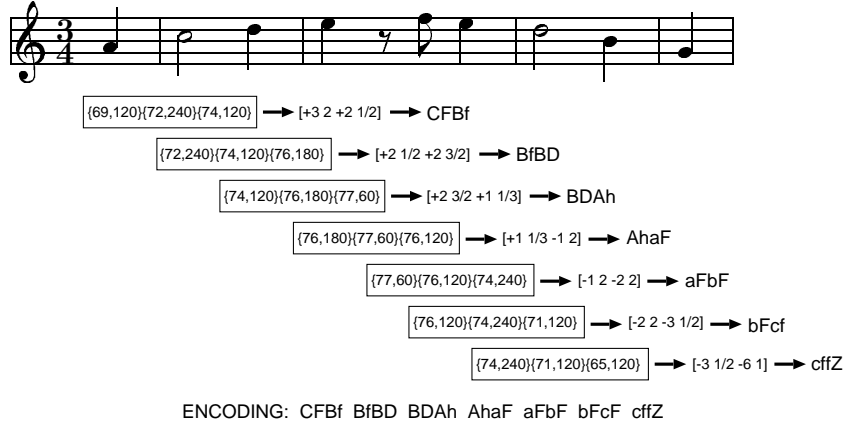
*$n$ -words coding.* The  $n$ -words obtained in the previous step are mapped into sequences of alphanumeric characters, which we will name  $n$ -words due to the equivalence we want to establish between melody and text. As the number of ASCII printable characters is lower than all possible intervals, a non-linear mapping is used to assign characters to different interval values (see [7] for details).

*Stop words.* A melody can contain pairs of notes separated by long rests, that can last for some seconds, or even minutes. Consecutive notes separated by this kind of rests are not really related, so the next note can be considered as the beginning of a new melody. Therefore, it is not fair encoding together consecutive notes separated by a large rest in a melody.

Considering this, a silence threshold is established, in a way that when a rest longer than this threshold is found, no words are generated across it. This restriction implies that, for each rest longer than this threshold,  $n-1$  words less are encoded. This threshold has been empirically set to a rest of four beats.

## 2.3 Word lengths

In order to test the classification ability of different word lengths,  $n$ , a range for  $n \in \{2, 3, 4, 5, 6, 7\}$  has been established. The shorter  $n$ -words are less specific and provide more general information and, on the other hand, larger  $n$ -words may be more informative but the models based on them will be more difficult to train. Using the encoding scheme, the vocabulary size for each  $n$  is  $|\mathcal{V}_n| = (53 \times 21)^{n-1}$  words. In Table 1 the number of words that have been extracted from the training set for each length is displayed.



**Fig. 1.** Example of a 3-word encoding of a MIDI file with 120 ticks per beat resolution. Sequences of pairs  $\{MIDI\ pitch, duration\ in\ ticks\}$  are extracted using a window of length 3. Then, intervals and IOR within the window are calculated and finally are encoded using the encoding scheme.

n	Corpus C1				Corpus C2				
	Jazz	Clas.	$ \mathcal{V}_n $	Coverage (%)	Bach	Greg.	Joplin	$ \mathcal{V}_n $	Coverage (%)
2	425	485	548	49,24	245	87	223	301	27,04
3	4883	3840	7903	0,64	1374	509	1471	2605	0,21
4	6481	6209	12501	$9,07 \cdot 10^{-4}$	2574	1207	2708	5835	$4,23 \cdot 10^{-4}$
5	6849	7390	14198	$9,25 \cdot 10^{-7}$	3245	1901	3245	8073	$5,26 \cdot 10^{-7}$
6	6967	8060	15013	$8,79 \cdot 10^{-10}$	3594	2297	3420	9165	$5,37 \cdot 10^{-10}$
7	7018	8483	15499	$8,15 \cdot 10^{-13}$	3728	2413	3463	9564	$5,03 \cdot 10^{-13}$

**Table 1.** Number of words in the training sets for the different word lengths: number of different words in each style, total of different words found in each corpus, and coverage of the vocabulary (percentage).

## 2.4 Naive Bayes Classifier

The naive Bayes classifier, as described in [8], has been used. In this framework, classification is performed following the well-known *Bayes' classification rule*. In a context where we have a set of classes  $c_j \in \mathcal{C} = \{c_1, c_2, \dots, c_{|\mathcal{C}|}\}$ , a melody  $x_i$  is assigned to the class  $c_j$  with maximum a posteriori probability, in order to minimize the probability of error:

$$P(c_j|x_i) = \frac{P(c_j)P(x_i|c_j)}{P(x_i)}. \quad (3)$$

Our classifier is based on the *naive Bayes assumption*, i.e. it assumes that all words in a melody are independent of each other, and also independent of the order they are generated. This assumption is clearly false in our problem and

also in the case of text classification, but naive Bayes can obtain near optimal classification errors in spite of that [9]. To reflect this independence assumption, melodies can be represented as a vector  $x_i = (x_{i1}, x_{i2}, \dots, x_{i|\mathcal{V}|})$ , where each component  $x_{it} \in \{0, 1\}$  represents whether the word  $w_t$  appears in the document or not, and  $|\mathcal{V}|$  is the size of the vocabulary. Thus, the class-conditional probability of a document  $P(x_i|c_j)$  is given by the probability distribution of words  $w_t$  in class  $c_j$ , which can be learned from a labelled training sample,  $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$ , using a supervised learning method.

**Multivariate Bernoulli model (MB)** Using this approach, each class follows a multivariate Bernoulli distribution:

$$P(x_i|c_j) = \prod_{t=1}^{|\mathcal{V}|} x_{it}P(w_t|c_j) + (1 - x_{it})(1 - P(w_t|c_j)) \quad (4)$$

where  $P(w_t|c_j)$  are the class-conditional probabilities of each word in the vocabulary, and these are the parameters to be learned from the training sample.

Bayes-optimal estimates for probabilities  $P(w_t|c_j)$ , with a Laplacean prior to smooth probabilities, and prior probabilities for classes  $P(c_j)$ , are calculated as:

$$P(w_t|c_j) = \frac{1 + M_{tj}}{2 + M_j} \quad P(c_j) = \frac{M_j}{|\mathcal{X}|} \quad (5)$$

where  $M_{tj}$  is the number of melodies in class  $c_j$  containing word  $w_t$ , and  $M_j$  is the total number of melodies in class  $c_j$ .

Classification of new melodies is performed then using Eq. 3, which is expanded using Eqs. 4 and 5.

**Multinomial model (MN)** This model takes into account word frequencies in each melody, rather than just the occurrence or non-occurrence of words as in the MB model. In consequence, documents are represented by a vector, where each component  $x_{it}$  is the number of occurrences of word  $w_t$  in the melody. In this model, the probability that a melody has been generated by a class  $c_j$  is the multinomial distribution, assuming that the melody length in words,  $|x_i|$ , is class-independent [8]:

$$P(x_i|c_j) = P(|x_i|)|x_i|! \prod_{t=1}^{|\mathcal{V}|} \frac{P(w_t|c_j)^{x_{it}}}{x_{it}!} \quad (6)$$

Now, Bayes-optimal estimates for class-conditional word probabilities are:

$$P(w_t|c_j) = \frac{1 + N_{tj}}{|\mathcal{V}| + \sum_{k=1}^{|\mathcal{V}|} N_{kj}} \quad (7)$$

where  $N_{tj}$  is the sum of occurrences of word  $w_t$  in melodies in class  $c_j$ . Class prior probabilities are also calculated as for MB.

**Multivariate Bernoulli mixture model (MMB)** Both MB and MN have proven to achieve quite good results in text classification [8,10], but they can be improved assuming that probabilities of words follow a more complex distribution within a class. Imagine that we want to model the distribution of *musical words* (see 2.2) in a sample of classical music melodies, which is formed in equal shares by Baroque and Renaissance scores. It is likely that both subsets have different distributions of words, so that we could find some words very common in Baroque music that don't usually appear in Renaissance music. In this case, using the MB model, Bayes-optimal estimates for these words in the whole set would be just a half of the real ones in Baroque, and much greater than their real value in Renaissance.

Intuitively, it would be more accurate to find the separate estimates for each substyle and then combine them to model the whole style. However, since the only information we have about these sequences is that all of them belong to classical music, finding the optimal estimates for each substyle is not straightforward. Furthermore, this problem becomes more complex when inner class structure (i.e. the number and proportions of the subsets) is not known a priori. To solve this problem, we will use here finite mixtures [11] of multivariate Bernoulli distributions, as they have been successfully applied in text classification tasks [12,10].

A finite mixture model is a probability distribution formed by a number of components  $M$ :

$$P(x_i) = \sum_{m=1}^M \pi_m P(x_i|p_m) \quad (8)$$

where  $\pi_m$  are the mixing proportions, that must satisfy the restriction  $\sum_{m=1}^M \pi_m = 1$ ; and  $p_m = (p_{m1}, p_{m2}, \dots, p_{m|V|})$  are the component prototypes. Since we will model each class as a mixture of multivariate Bernoulli distributions, each component distribution  $P(x_i|p_m)$  is calculated using Eq. 4, substituting  $P(w_i|c_j)$  with its corresponding value  $p_{mt}$ .

Now we face the problem of obtaining optimal estimates for parameters  $\Theta = (\pi_1, \dots, \pi_M, p_1, \dots, p_M)^t$ . This can be achieved using the EM algorithm [13], but to be applicable, EM algorithm requires that the problem be formulated as an incomplete-data problem. To do this, we can think of each sample document  $x_i$  as an incomplete vector [12], where  $z_i = (z_{i1}, \dots, z_{iM})$  is the *missing data* and indicates which component of the mixture the document belongs to (with 1 in the position corresponding to the component and zeros elsewhere). Then, the EM proceeds iteratively to find the parameters that maximize the log-likelihood of the complete data:

$$\mathcal{L}_C(\Theta|X, Z) = \sum_{i=1}^n \sum_{m=1}^M z_{im} (\log \pi_m + \log P(x_i|p_m)). \quad (9)$$

## 2.5 Feature selection

The methods explained above use a representation of musical pieces as a vector of symbols. A common practice in text classification is to reduce the dimensionality

of those vectors by selecting the words which contribute most to discriminate the class of a document. A widely used measure to rank the words is the *average mutual information* (AMI) [14].

For the MB model, the AMI is calculated between (1) the class of a document and (2) the absence or presence of a word in the document. We define  $C$  as a random variable over all classes, and  $F_t$  as a random variable over the absence or presence of word  $w_t$  in a melody,  $F_t$  taking on values in  $f_t \in \{0, 1\}$ , where  $f_t = 0$  indicates the absence of word  $w_t$  and  $f_t = 1$  indicates the presence of word  $w_t$ . The AMI is calculated for each  $w_t$  as<sup>1</sup>:

$$I(C; F_t) = \sum_{j=1}^{|C|} \sum_{f_t \in \{0,1\}} P(c_j, f_t) \log \frac{P(c_j, f_t)}{P(c_j)P(f_t)} \quad (10)$$

where  $P(c_j)$  is the number of melodies for class  $c_j$  divided by the total number of melodies;  $P(f_t)$  is the number of melodies containing the word  $w_t$  divided by the total number of melodies; and  $P(c_j, f_t)$  is the number of melodies in class  $c_j$  having a value  $f_t$  for word  $w_t$  divided by the total number of melodies.

In the MN model, the AMI is calculated between (1) the class of the melody from which a word occurrence is drawn and (2) a random variable over all the word occurrences, instead of melodies. In this case, Eq. 10 is also used, but  $P(c_j)$  is the number of word occurrences appearing in melodies in class  $c_j$  divided by the total number of word occurrences,  $P(f_t)$  is the number of occurrences of the word  $w_t$  divided by the total number of word occurrences, and  $P(c_j, f_t)$  is the number of occurrences of word  $w_t$  in melodies with class label  $c_j$ , divided by the total number of word occurrences.

### 3 Results

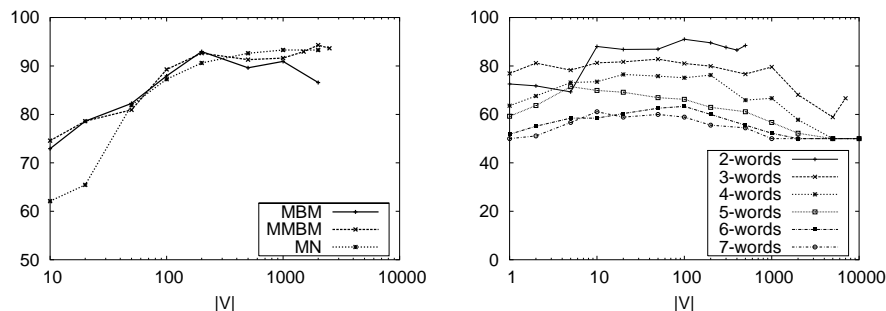
For each model and word size, the naive Bayes classifier was applied to the  $n$ -words extracted from the melodies in our training set in order to test its style recognition ability. The experiments have been made following a leave-one-out scheme. The presented results are the percentage of successfully classified melodies.

The evolution of the classification as a function of the significance of the information used is presented in the graphs in Fig. 2. For this, the words in the training set have been ordered according to their AMI value. Also, experiments using only the best rated words ( $|\mathcal{V}|$  in the graphs) have been performed.

Note that the results were not conclusive in terms of the different statistical models, since all the methods performed comparatively. There is a tendency of Bernoullis to classify better for small values of  $|\mathcal{V}|$  while multinomials seem to provide better results for larger  $|\mathcal{V}|$ .

Table 2 shows the best results obtained in the experiments. The best accuracy was obtained for the word size  $n = 3$ , reaching a 94.3% of successful style identification. Large  $n$ -words only perform well for very small  $|\mathcal{V}|$  values, and

<sup>1</sup> The convention  $0 \log 0 = 0$  was used, since  $x \log x \rightarrow 0$  as  $x \rightarrow 0$ .



**Fig. 2.** Evolution of style recognition percentage in average for different vocabulary sizes. The plots represent **(left)** a comparison of the different statistical models (displayed for corpus C1) and **(right)** word sizes (for corpus C2).

get worse rapidly for larger values. This preference for little specific information points to the fact that the training set is small for those lengths, and the results could be improved for larger models with more training melodies.

Also the values for precision and recall have been studied. Recall figures get very low as  $n$  increases, being the cause of the lower classification rates obtained for large words. In fact, the tendency for lengths  $n = 2, 3$  is to get low percentage rates when  $|\mathcal{V}|$  increases due to low recall and very high precision values: there are a lot of unclassified melodies, but the decisions taken by the classifier are usually very precise. It can be said that the classifier learns very well but little. This fact also reflects the need of a larger training set for these lengths to be successfully applied.

n	Corpus C1					Corpus C2				
	% ok	model	$ \mathcal{V} $	f	M	% ok	model	$ \mathcal{V} $	f	M
2	92.1	MMB	100	2	2	90.6	MN	100	2	-
3	89.1	MMB	7500	2	4	94.3	MMB	2000	2	3
4	80.7	MB	100	1	-	90.6	MB	1000	1	-
5	82.3	MN	200	1	-	84.6	MN	7000	1	-
6	78.8	MN	14000	1	-	69.5	MN	50	1	-
7	73.9	MN	10000	1	-	53.3	MB	500	1	-

**Table 2.** Best results in classification percentages obtained for both corpora. For each word length value,  $n$ , the table shows, from left to right: best classification, statistical model, size of vocabulary used for it, encoding scheme and number of components (for the mixture model).

We have compared our results to those obtained by our research group with corpus C1, using melodic, harmonic and rhythmic statistical descriptors. They are fed into supervised classifiers like nearest neighbours (NN) or a standard Bayes rule [5]. In those experiments, the best recognition rates obtained for whole melodies were 91.0% for Bayes and 93.0% for NN, after a long study of the parameter space and the descriptor selection procedures. Thus, the first results obtained under this new approach are very encouraging.

## 4 Conclusions

The feasibility of using text categorization technologies for music style recognition has been tested. The models based on 2-words had the best performance, although the best result obtained was for  $n = 3$ , reaching a 94.3% of successful style recognition. Larger word lengths have provided also good results using small vocabulary sizes. In these cases, the method has proved to be very accurate, but lacks retrieval power. This fact points to a lack of training data. The various statistical models tested did not present significant differences in classification.

The results have been compared to those obtained by other description and classification techniques, providing similar or even better results. We are convinced that an increment of the data available for training will improve the results clearly, specially for larger  $n$ -word sizes.

**Acknowledgment:** This work was supported by the projects Spanish CICYT TIC2003–08496–C04, EU ERDF funds, and Generalitat Valenciana GV043–541.

## References

1. Zhu, J., Xue, X., Lu, H.: Musical genre classification by instrumental features. In: Int. Computer Music Conference, ICMC 2004. (2004) 580–583
2. Whitman, B., Flake, G., Lawrence, S.: Artist detection in music with minnow-match. In: Proc. of 2001 IEEE Workshop on Neural Networks for Signal Processing. (2001) 559–568
3. Soltau, H., Schultz, T., Westphal, M., Waibel, A.: Recognition of music types. In: Proc. of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP-1998). (1998)
4. Cruz, P.P., Vidal, E., Pérez-Cortes, J.C.: Musical style identification using grammatical inference: The encoding problem. In Sanfeliu, A., Ruiz-Shulcloper, J., eds.: Proc. of CIARP 2003. (2003) 375–382
5. Ponce de León, P.J., Iñesta, J.M.: Feature-driven recognition of music styles. In: 1st Iberian Conference on Pattern Recognition and Image Analysis. LNCS, 2652. (2003) 773–781
6. McKay, C., Fujinaga, I.: Automatic genre classification using large high-level musical feature sets. In: Int. C. Music Information Retrieval, ISMIR'04. (2004) 525–530
7. Doraisamy, S., Rüger, S.: Robust polyphonic music retrieval with n-grams. *Journal of Intelligent Information Systems* **21** (2003) 53–70
8. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. In: AAAI-98 W. on Learning for Text Categorization. (1998) 41–48
9. Domingos, P., Pazzani, M.: Beyond independence: conditions for the optimality of simple bayesian classifier. *Machine Learning* **29** (1997) 103–130
10. Novovičová, J., Malík, A.: Text document classification using finite mixtures. Technical Report 2063, Academy of Sciences of the Czech Republic, Institute of Information Theory and Automation (2002)
11. McLachlan, G., Peel, D.: *Finite Mixture Models*. John Wiley & Sons (2000)
12. Juan, A., Vidal, E.: On the use of Bernoulli mixture models for text classification. *Pattern Recognition* **35** (2002) 2705–2710
13. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. *J. of the Royal Statistical Society B* **39** (1977) 1–38
14. Cover, T.M., Thomas, J.A.: *Elements of Information Theory*. John Wiley (1991)