Compressing Lists for Audio Classification

Teppo E. Ahonen
Department of Computer Science
University of Helsinki
Helsinki, Finland
teahonen@cs.helsinki.fi

ABSTRACT

Normalized Compression Distance (NCD) is an information-theory based similarity metric that has been used successfully for similarity measuring in various domains, including music. Here, we extend NCD from the pairwise similarity measurement to lists of objects. Based on the compressibility of a single object in the context of a list, we can make assumptions of the objects similarity with the objects in the given list.

We apply the list compression into the task of classifying music in audio format according to the genre. We use features derived from a set of audio data as the list representations of a genre. Then, for a single piece of music that needs to be classified, we extract the same features and measure how well the representation compresses with a list of same features from different pieces of music. We present three variants of practical implementations of the system.

The evaluation shows that our approach has potential. Preliminary results seem promising and suggest that the method can be extended by applying more features into the measurement process. In addition, using different features allows utilizing the method for other classification tasks.

Categories and Subject Descriptors

J.5 [**Computer Applications**]: Music

General Terms

Algorithms, Theory

Keywords

audio genre classification, normalized compression distance

1. INTRODUCTION

Automatic classification is one of the most common tasks in Music Information Retrieval (MIR). Music can be classified based on various different criteria such as genre, composer or mood. Especially genre classification has been studied extensively in the recent years, with a plethora of different approaches and methods. For a tutorial on genre classification, we refer to [6].

The basis of classifying audio music is extracting relevant features from the signal data and then applying some machine learning technique on the features. Common approaches include Support Vector Machines (SVM), Gaussian Mixture Models and k-Nearest Neighbor (k-NN) classification.

Normalized Compression Distance (NCD) [2] is a distance metric that has its roots in information theory. In NCD, the distance between two objects is measured using data compression; the better the two objects compress together, the more information they share and thus, the more similar they are. NCD has been used successfully for clustering and classification tasks in various different domains, including music. In audio genre classification, there are to our knowledge two NCD-based approaches. In [4], the dictionary generated by the LZ78 algorithm was used as a SVM string kernel, and in [1], NCD was utilized for training a k-NN classifier with MIDI files in conjunction with audio data.

Here, NCD is extended from pairwise distance into measuring similarity in a list of objects. The method is applied for genre classification: we extract MFCC vectors from the audio data, then quantize the vectors into a discrete alphabet and use data compression to determine the similarity of a single object with a list of lexicographically sorted objects in length-increasing order. The single object is then classified according to the list that it was most similar with.

The rest of the paper is organized as follows. In Section 2, we present the theory and concept of NCD and its extension to lists. In Section 3, we present our approach for classifying audio data by genre using list compression and report results for a 1000 piece of music data set classification. Finally, conclusions and future work are presented in Section 4.

2. NORMALIZED COMPRESSION DISTANCE

Kolmogorov complexity of an object x is the length of the shortest prefix-free binary program that produces x as an output, denoted here $K(x)$. Also, conditional Kolmogorov, meaning the length of the shortest binary program that produces x as an output when given y as an input, is denoted $K(x|y)$.

Using Kolmogorov complexity, a universal similarity metric called Normalized Information Distance (NID) has been defined. NID for two objects x and y is denoted [2]

$$NID(x,y) = \frac{max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}. \tag{1}$$

The NID is a metric, and also universal [2]. However, Kolmogorov complexity is uncomputable, making NID also uncomputable. But Kolmogorov complexity can be approximated using data compression. By approximating $K(x)$ with $C(x)$, where $C(x)$ is the length of the string x when compressed using a standard lossless data compression algorithm C, the NCD can now be denoted [2]

$$NCD(x,y) = \frac{C(xy) - min\{C(x), C(y)\}}{max\{C(x), C(y)\}}, \tag{2}$$

where xy is the concatenation of x and y. The NCD is a metric, and it is also quasi-universal, depending on how well the compressor approximates the Kolmogorov complexity [2].

2.1 Compressing Finite Lists

Pairwise distance measuring can be applied for various tasks, but it would seem that a way to measure information shared in many objects could be a more practical tool for various tasks. So far, this approach has not been studied extensively. In [5], the concept of similarity in multiple objects was used for mining specialized reviews. They used the size of a set of words as an approximation of the Kolmogorov complexity.

In [7], the list information distance is presented. For a finite list $V = (x_1, \ldots, x_m)$, in length-increasing lexicographic order, the information distance given string x_i to any string in V is $E_{max}(V)$, and in [5] it is shown to be

$$E_{max}(V) = \max_{x:x\in V} K(V|x). \tag{3}$$

Thus, $E_{max}(V)$ is the smallest amount of information needed to convert any string in V to any other string in V. The $E_{max}(V)$ is universal and a metric [7].

Also, in [7], E_{min} is defined as "the program length of the most comprehensive object that contains the most information about all the other elements of V", and formally as

$$E_{min}(V) = \min_{x:x\in V} K(V|x). \tag{4}$$

In order to be able to measure similarity between lists of various sizes, a normalization factor needs to be added to Equation (3). In [7], several proposals of a normalized list information distance are presented. Here, we use the one of the form

$$e(V) = \frac{K(V|x_V)}{K(V')} \tag{5}$$

where x_V is the element x in V that produces the $E_{\max}(V)$, and V' is V with x_V deleted.

It should be noted that the $e(V)$ as defined above is not a metric; a counter-example can be given for which the triangle inequality property of a metric does not hold [7]. We suggest that despite this, it can be used as a basis for classification.

For practical implementation of the normalized list information distance, we experimented with three versions. The first is of the form

$$e_1(V) = \frac{C(V) - C(x)}{C(V')} \tag{6}$$

where $C(V)$ is the length of the list V when compressed using data compression algorithm C, $C(x)$ is the same to the string x and V' is the list V with x deleted. The difference to Equation (5) is that instead of using x_V, which is the object in the list that yields the largest information distance for V, we use the candidate string representation for calculating the distance for the list that includes the candidate.

Thus, it is not a direct approximation of $e(V)$, but instead it measures the similarity between the candidate and the list. However, it gives an idea of how well a single object compresses when included to a list. In classification, we calculate $e_1(V)$ for all lists, and then classify x according to the lowest $e_1(V)$ value.

Another, probably closer variant we came up with is a measurement which measures the change in the $e(V)$ when an object is added to the list. We define this as the difference between the normalized list information distances with and without the candidate included in the list. We denote this

$$\Delta e(V) = \frac{K(V_g|x_{V_g})}{K(V_g')} - \frac{K(V|x_V)}{K(V')}, \tag{7}$$

where V_g is the list with the candidate vector included and V is the list without the candidate vector.

This describes the amount of change in information of the list when the candidate string is added. The non-metricity of the normalized list information distance is still an issue here. In practice, we approximate K with C analogously to that of Equation (6), and in classification, we classify x according to the smallest $\Delta e(V)$ value.

Third version utilizes a slightly different approach. From the list consisting of the example pieces of the genre, we look for an object that contains the most of the information of all the others in the list, that is, the object for which achieves $E_{\min}(V)$ of Equation (4). Then, we use the pairwise NCD of Equation (2) to calculate the similarity between the candidate and the representative of the genre. Thus, the candidate is classified according to the smallest NCD value. It should be noted that here the normalization does not take place when selecting the representative from the list, but when calculating the NCD value between the candidate and the representative.

3. PROPOSED GENRE CLASSIFICATION METHOD

We present an approach for genre classification that utilizes NCD for lists of objects as the classification method. As features, we use Mel-Frequency Cepstral Coefficient (MFCC) vectors. We calculate the first eight coefficients from the signal, with a window size of 23.2 milliseconds and 50 % overlap. For the audio feature extraction, we use MIRToolbox, version 1.3 [3].

To be able to compress the vectors reasonably, the continuous MFCC vectors need to be quantized and dimensionally reduced. We experimented with several quantization methods, but due to space constraints, present here results only for the method we had the highest precision with: k-means.

We clustered the MFCC vectors of the given piece of music and then represented the vectors as a sequence of cluster indexes. After experimenting with various values of k, we eventually settled on the value of 64.

We used a dataset that has been used for developing and testing various genre classification systems. The dataset consists of 1000 30-second samples, divided in 10 genres with 100 samples in each genre. It is also available online[1]. For the evaluation, we run a 10-fold cross validation, selecting randomly 10 samples from each genre as the training data, with no overlapping in the training data sets between the folds.

In training phase, the MFCC vectors from a set on genre examples are extracted and quantized, and then sorted into a length-increasing lexicographical order. The elements of the list are separated with a special character between each element when the list is written as a file. For compression, we use the bzip2 algorithm.

When classifying a piece of music, the MFCC vectors were extracted and quantized analogously to that of the MFCC vectors of the example pieces. Then the similarities between each genre and the candidate were calculated according to one of the methods described in Subsection 2.1 and the candidate was classified according to the smallest distance to a genre.

3.1 Results of the evaluation

The results of the classification based on Equation (6) are presented in Table 1. The most distinctive genres stand out in the confusion matrix: these are classical music (cl), hiphop (hi), jazz (ja) and metal (me), each having distinguishable instrument characteristics. The genres with much more common instrumentation (blues (bl), country (co), disco (di), pop (po), reggae (re) and rock (ro)), on the other hand, confuse. In average, the precision of the method is 0.490 and recall 0.449, making the average F-measure value 0.469.

Using Equation (7) as the classification method produces confusion matrix presented in Table 2. The values here do seem less distinctive than the values in Table 1, making it evident that measuring the amount of change in the information, though theoretically interesting, may not work well in practice for classification. It should be also noted that this method is very time-consuming in comparison to the other two approaches presented here. The average precision, recall and F-measure values for the method are 0.458, 0.423 and 0.440, respectively.

The third version, where we first selected a single piece of music as the representative of the genre and then calculated the representative's similarity to the candidate vector using NCD, produced confusion matrix depicted in Table 3. The values are more distinctive than the values in Table 2, but there seems to be more confusion between some genres than in the results depicted in Table 1. For this method, the average precision is 0.479, recall 0.433 and the F-measure 0.455. It it also worthwhile to notice that Tables 1 and 3 have several somewhat similar patterns, suggesting that $E_{min}(V)$ indeed provides information of the object that contains most of the information about all the other objects in the list.

3.2 Discussion

Based on the results depicted in Tables 1, 2 and 3, it seems that list compression provides a possible tool for audio clas-

Table 1: Confusion matrix for classification based on Equation (6).

		Ground Truth									
		Bl	Cl	Co	Di	Hi	Ja	Me	Po	Re	Ro
Classification	Bl	**27**	0	4	3	1	0	0	2	3	5
	Cl	2	**77**	8	0	0	24	0	2	0	4
	Co	13	0	**29**	28	5	9	7	21	14	28
	Di	2	0	0	**31**	18	0	12	8	6	0
	Hi	1	0	0	10	**61**	0	6	14	7	0
	Ja	18	23	23	1	0	**51**	3	0	3	6
	Me	3	0	0	0	0	8	**62**	0	3	4
	Po	15	0	11	19	9	0	8	**38**	22	19
	Re	2	0	2	8	4	0	0	5	**39**	0
	Ro	17	0	23	0	2	8	2	10	3	**34**

Table 2: Confusion matrix for classification based on Equation (7).

		Ground Truth									
		Bl	Cl	Co	Di	Hi	Ja	Me	Po	Re	Ro
Classification	Bl	**23**	0	1	2	0	0	2	6	1	0
	Cl	13	**92**	19	1	0	46	2	2	2	4
	Co	8	0	**26**	25	10	10	7	20	5	25
	Di	3	0	0	**36**	25	0	6	15	6	1
	Hi	0	0	0	3	**38**	0	8	8	8	1
	Ja	8	8	18	1	0	**28**	0	1	3	3
	Me	5	0	2	5	8	6	**66**	0	3	2
	Po	6	0	6	4	11	4	5	**33**	15	15
	Re	2	0	0	10	5	0	1	3	**40**	8
	Ro	32	0	28	13	3	6	3	12	17	**41**

Table 3: Confusion matrix for classification based on the pairwise NCD value between the candidate and the genre representative.

		Ground Truth									
		Bl	Cl	Co	Di	Hi	Ja	Me	Po	Re	Ro
Classification	Bl	**26**	0	3	3	1	0	0	2	3	5
	Cl	2	**76**	8	0	0	24	0	2	0	4
	Co	13	0	**28**	29	5	9	7	21	14	29
	Di	2	0	0	**29**	17	1	12	8	6	0
	Hi	1	0	0	10	**60**	0	6	14	8	0
	Ja	18	23	25	1	0	**44**	3	0	4	6
	Me	3	0	0	0	1	8	**62**	0	3	4
	Po	16	0	11	19	9	5	8	**38**	22	19
	Re	2	0	2	6	4	0	0	5	**37**	0
	Ro	17	1	23	3	3	9	2	10	3	**33**

[1]http://opihi.cs.uvic.ca/sound/genres/

sification. All in all, all three suggested versions provided very similar results with only minor differences in the confusion matrices and there are very little differences in their precision, recall and F-measure values (see Table 4 for comparison). The best results were obtained with Equation (6), which is a very straightforward list adaptation of the pairwise NCD.

There are some clear hindrances that have impact on the results. The confusion between genres with less distinguishable instrumentation (especially blues, country and rock) suggests that the MFCC vectors may not contain enough information for accurate classification. Thus, several other features, such as beat histograms or chroma features, could be added to the classification process. Also, the quantization process of the MFCC vectors used here may not be the optimal solution for the task.

4. CONCLUSIONS

We have presented a new method for classifying audio data by genre. As a variant of the common "train and test" -approach, we use selected audio files as an example set of an genre, and classify new pieces of music according to how well feature representations derived from them compress together with the representations derived from the example pieces of music. Thus, our approach is very straightforward; there is no need for a difficult and possibly tedious training of a classifier, but in the training phase, all it takes is to create lists from the features of the training data.

We presented three different approaches to use list compression for classification. The first considered the amount of information contained in the candidate vector in comparison to the rest of the list, the second the change in information contained in the list when the candidate is added to the list and the third the normalized compression distance between the most representative object in the list and the candidate. In our experiments, the first version provided slightly better results than the others.

Based on the results, the approach seems to have potential and the list compression can be utilized as a possible method for classification tasks. The method, however, does not reach the classification accuracy of the another compression-based method proposed in [4] using the same test data. This implies that several open issues need to be considered.

4.1 Future work

The work presented here has been preliminary and several issues presented in this paper need to be examined more carefully in order to be able to present a valid and justifiable version of an audio classifier based on list compression.

The three presented measuring techniques presented here are all based on empirical testing and their theoretical backgrounds are not discussed yet. In future, we will address the theoretical aspects of the proposed measuring techniques and eventually come up with a more well-grounded measuring technique based on list compression. Also, the fact that the normalized list information distance we build our work on here is not a metric is somewhat unsettling.

Table 4: Average precision, recall and F-measures and standard deviations for the different classification methods.

Method	Precision	Recall	F-measure	STDev
$e_1(V)$	0.490	0.449	0.469	16.954
$\Delta e(V)$	0.458	0.423	0.440	21.151
Pairwise	0.479	0.433	0.455	17.030

We have used only one feature derived from the audio data. Successful classification could benefit from using more features, and by using various different features, we would not be limited to only genre classification, but another kind of classifications could be done, for example audio classification based on mood or composer.

We feel also that the collection of 30-second audio snippets can be too limited and impractical for compression-based similarity measurement. Keeping in mind that the difference between Kolmogorov complexity K and compression-based approximation C minimizes as the length of x increases [2], longer sequences could provide more accurate classification results. In the future, we will experiment our approach with a large set of complete pieces of music.

We are also considering experimenting our approach with symbolic music data. With symbolic music, the process of translating data into string representations is more straightforward and thus the harm caused by discretization is a lesser issue.

5. ACKNOWLEDGMENTS

This work was supported by the Academy of Finland, grant #129909.

6. REFERENCES

[1] Z. Cataltepe, Y. Yaslan, and A. Sonmez. Music genre classification using MIDI and audio features. *EURASIP Journal on Applied Signal Processing*, 2007(1), 2007.

[2] R. Cilibrasi and P. M. B. Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, 2005.

[3] O. Lartillot and P. Toiviainen. A MATLAB toolbox for music feature extraction from audio. In *Proceedings of the 10th International Conference on Digital Audio Effects*, Bordeaux, France, 2007.

[4] M. Li and R. Sleep. Genre classification via an LZ78-based string kernel. In *ISMIR'05*, London, UK, 2005.

[5] C. Long, X. Zhu, M. Li, and B. Ma. Information shared by many objects. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, New York, NY, USA, 2008. ACM.

[6] N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: a survey. *Signal Processing Magazine, IEEE*, 23(2):133–141, 2006.

[7] P. M. B. Vitányi. Information distance in multiples. Submitted. Preprint available: arXiv:0905.3347v1.