MELODY CLASSIFICATION USING A SIMILARITY METRIC BASED ON KOLMOGOROV COMPLEXITY

Ming Li and Ronan Sleep
School of Computing Sciences, UEA, Norwich NR47TJ, UK
mli, mrs@cmp.uea.ac.uk

ABSTRACT

Vitanyi and his co-workers [5] have reported some success using a universal similarity metric based on Kolmogorov complexity for a variety of classification tasks, including music genre recognition. This paper describes new experiments in this direction, and compares the results with some alternative approaches. Somewhat to our surprise given its non-specific universal nature, the Kolmogorov complexity similarity based technique outperformed the others.

The task used for our experiments involved classification of MIDI files into one of 4 groups. Two of the categories were western classical music composed by Beethoven (302 files) and Haydn (261 files). The remaining categories were Chinese music (80 files) and Jazz (128 files). Melody contours (i.e. pitch sequences without timing details) were extracted from the MIDI file tracks. Both relative and absolute and pitch contours were used.

The best performance of 92.35% was achieved by a 1-nearest neighbour classifier with normalized information distance based on Kolmogorov complexity estimates over pitch interval contours. A notable feature of our work is the use of the number of blocks in a pure Lempel-Zip parse of a string to estimate its Kolmogorov complexity.

1. INTRODUCTION

Most people can hum at least a fragment of a familiar tune, providing the basis for *query by humming* music retrieval systems. Much of the information about the original music is lost in the hummed version, but features preserved include the order in which the notes are played, their contour, and timing. For the best classification performance, all three characteristics can be used, with timing seeming to be of particular importance in identifying music genre. However, it is interesting to note that the Dictionary of Musical Themes [12] is based solely on the note sequence, ignoring timing information entirely . Parson's Directory of Tunes and Musical Themes [11], known as the 'UP DOWN book', takes this further, retaining only directional information about pitch change. This suggests that we can index *themes* without timing information. In this paper we explore the possibility of indexing *genre* as opposed to theme, without using timing information. At the same time we explore the use of a universal similarity metric to guide the classification.

Given only the absolute pitch (or pitch interval) sequence within the melody, there remains a wealth of methods used for classification. *Ensemble statistics* methods are based on pitch frequency, ignoring sequence information. Examples include [1] and [2]. *Sequence based methods* exploit information about note ordering.

Independently of whether or not they use information about note ordering, methods can be further divided into two sub-categories: *model-based* and *similarity measure* methods:

- In a model based method, the sequence is assumed to be generated by some underlying probabilistic model (e.g. Hidden Markov Model [3] or other Language Model [4]) for each class. The classification is achieved by estimating the probability for each class during a training phase, and associating a sequence with the model that gives the best explanation of the sequence.

- Similarity measure methods rely on some measure of distance between two objects. For example, one well known measure of distance between string objects is the edit distance, defined as the minimal number of primitive edit operations required to transform one string into another. With such a measure, a classifier such as k-nearest neighbour (k-NN) can be used.

In this paper we adopt the the similarity measure approach. We present some preliminary results designed to explore the effectiveness of a new similarly measure based on the concept of Kolmogorov complexity [5]. Informally, the Kolmogorov complexity of an object is the size of the smallest program which can be used to generate the object. This is an idealized notion, because it is not computable. However, any compression algorithm gives an upper bound and this can be taken as an estimate of the Kolmogorov complexity. Following [5], we use our approximation to construct similarity measures which place two objects near each other if we can significantly compress one given the information in the other.

Such a similarity metric (called a *K-metric* below) amounts to relative compressibility. It is attractive because it does not rely on any detailed knowledge about the nature of the objects: it is in this sense that it is *universal*. But the very universality of the measure suggests that it is not going to perform very well when used to drive a classifier precisely because it ignores domain specific knowledge. The work reported below suggests either that this intuition may be wrong or alternatively that attempts to build knowledge based classifiers could do better.

The remainder of the paper is organized as follows. Section 2 introduces a normalized distance metric based on Kolmogorov Complexity. Section 3 gives details of the compression algorithm we used for complexity estimation. Details of the classification experiments and results are shown in section 4, and the paper is concluded in section 5.

2. MELODY SIMILARITY AND INFORMATION-BASED SEQUENCE DISTANCE

Using a similarity based metric based on compressibility allows us to estimate how much information is shared between two pieces of music. There is no guarantee that this will be useful for genre classification, but it *might* be. Intuitively, when we mentally arrange a number of pieces of music in our heads we are likely to take advantage of any shared structure. If our human notions of musical genre have a relation to this shared structure, and if it is possible to detect degrees of structure sharing using relative compressibility, then the use of a K-metric will have some degree of success in classification. Other authors have made similar suggestions: for example, [13] suggests that one way of telling that a musical style is similar to another one, the second of which you already recognized, is that the first of the styles also appeals to you.

2.1. Conditional Kolmogorov Complexity

Given an object encoded as a binary string x, its *Kolmogorov complexity [16]* $K(x)$ is defined as the minimum number of bits into which the string can be compressed without losing information. Intuitively, Kolmogorov complexity indicates the descriptive complexity contained in an object and it is equal to the length of the shortest program for some universal machine which, when run without any input, outputs that string. A random string has relatively high complexity since no structural pattern can be recognized to help reduce the size of program. Strings like melody sequences should have lower complexity due to repeated phrases and musical structure.

For our music application, objects will be sequences of descriptive symbols. Given two sequences x and y, the conditional Kolmogorov complexity $K(x|y)$ is defined as the shortest program that can regenerate x from y. $K(x)$ is the special case $K(x|\lambda)$ where λ is the empty sequence.

2.2. Normalized Information Distance

The *information distance* [17] between two sequences x and y can be defined as the length of a shortest binary program that computes x given y, and also computes y given x. However, such a distance does not take the length of the sequence into account. We do not want to classify a very short piece of classical music as jazz simply because our examples of classical music are long whilst most of our jazz examples are short. This motivates the desire for a relative measure that takes account of sequence length. If two pairs of sequences have the same K-metric distance but with different lengths, the longer pair should be given a smaller distance measure than the shorter pair, reflecting the fact that more information is shared between longer sequences.

The authors in [5] propose a normalized information distance $D(x, y)$ for measuring the similarity relations between sequences. Such a distance takes values in $[0, 1]$ for all sequence pairs and should satisfy the following relations:

1. $D(x, y) = 0$ iff $x = y$
2. $D(x, y) = D(y, x)$ (symmetry)
3. $D(x, y) <= D(x, z)+D(z, y)$ (triangle inequality)

Two versions of the similarity metric are proposed in [5].

$$d_1(x, y) = \frac{K(x|y) + K(y|x)}{K(xy)} \tag{1}$$

$$d_2(x, y) = \frac{max(K(x|y), K(y|x))}{max(K(x)), max(K(y))} \tag{2}$$

The second definition can be shown to satisfy the conditions enumerated above without qualification, and in that sense is more satisfactory than the first. However, because conditional Kolmogorov complexity is not computable, we have to rely on estimates of K. This makes it less clear that direct application of this mathematical elegance criterion is the right thing to do. We used both variants in our experiments. Note that with either metric, the more information that is shared between two sequences, the less will be the distance measure.

Both definitions require estimates of the conditional complexity $K(x|y)$. We followed [5] in estimating $K(x|y)$ as the difference of the unconditional complexity estimates $\hat{K}(xy)$ and $\hat{K}(y)$:

$$\hat{K}(x|y) = \hat{K}(xy) - \hat{K}(y) \tag{3}$$

Here xy stands for the concatenation of sequences x and y. In general, the order of concatenation affects the size of the compressed concatenation, so that the relation $\hat{K}(xy) = \hat{K}(yx)$ may not hold. We dealt with this issue by using the average of the two orderings.

3. ESTIMATION OF UNCONDITIONAL KOLMOGOROV COMPLEXITY

We now turn to the problem of finding the unconditional Kolmogorov complexity of an object. Due to its incomputable nature this can only approximated. [5] suggest the use of industrial compressors such as gzip to do this. We deviate from [5] by using as our estimator of Kolmogorov complexity the number of blocks emitted by what is arguably the simplest and most elegant of all dictionary compression algorithms: the LZ78 parsing algorithm [7]. In fact we used the LZW variant [8] to obtain our best results, but the differences in performance between the two are not great, and we had found the pure LZ78 algorithm is easier to grasp at a first reading. Recall that λ denotes the empty string:

```
/* count LZ78 segments */
Clear dictionary;
w = $\lambda$;
Kest = 0;

while (more input)
      C = next symbol;
      if (wC in dictionary)
         w = wC;
      else
         Kest = Kest+1;
         add wC to dictionary
         w = $\lambda$;
      endif
endwhile

if (w!=$\lambda$)
      Kest = Kest+1;
endif

return Kest;
```

This captures the essence of a Lempel-Ziv encoder: at each step it examines the incoming symbol stream and consumes the largest substring stored in the dictionary. It then creates a new dictionary entry consisting of the longest matching entry just found, followed by the next symbol in the input. The full algorithm then emits a codeword consisting of a compact reference to the new entry. As we are only interested in counting the number of codewords output, we omit this coding step and simply increment variable $Kest$ at each new dictionary entry. The final value of $Kest$ is the number of blocks in an LZ78-parse of the input. $Kest$ is taken as our estimate $\hat{K}$.

Notice that our K_{est} is not necessarily an upper bound on the Kolmogorov complexity. For example, we ignore the number of bits required to represent the codewords for each block. However, the length of an LZ78 parse does give a simple clear measure of object size uncluttered by details such as code window sizes that bedevil practical compressors. Its clarity and simplicity provide a precise definition for experimental use, in contrast to the use of unspecified versions of standard compressors. Further, LZ78 it is extremely fast, and its theoretical properties have been extensively studied. This increases the chances that its properties with regard to source separation and classification might be amenable to theoretical analysis. Of course we have departed significantly from the Vitanyi model in our use of LZ78 parse length as an estimator, so any results we obtain - either practical or theoretical - may not be valid in his more general setting.

4. EXPERIMENTAL METHODOLOGY AND RESULTS

4.1. Music representation and Dataset

771 MIDI files falling into 4 categories were collected from the internet. Two of the categories were western classical music composed by Beethoven (302 files) and Haydn (261 files). The remaining categories were Chinese music (80 files) and Jazz (128 files). We are aware that our dataset is unbalanced, and this needs to be borne in mind when interpreting our results.

Melodies represented by sequence of both absolute pitches and pitch intervals were extracted from the MIDI files as follows. First, a monophonic melody was obtained by combining all the tracks except percussion ones. If two note-on events happened simultaneously, only the one with highest pitch value was preserved. We then removed information about the onset time and duration of each note, retaining only the ordering of the notes. The aim of this preprocessing was to extract the essence of the melody from the MIDI file, removing unintended clues and MIDI formatting clutter. It also makes the results of the experiments described here directly comparable with other classification experiments carried out by the authors [18].

We used a k-NN classifier to test the effectiveness of our K-measure. This was compared with a statistical model based on tri-grams, in which all the tri-grams are used for model training. During the test, if an unseen event appears, Katz's backing-off model is used to smooth the probability estimation. We also used a second point of comparison, based on using trigram distributions to drive a multi-class Support Vector Machine (SVM). We used SVMTorch [9] in this work and the entire configuration is used as default.

A standard three-fold stratified cross validation was carried out to evaluate classification performance.

4.2. Experimental Results and Discussion

TABLE 1: Performance figures

Classifier	Feature	Relative Pitch	Absolute Pitch
1-NN	d_1	92.4 (1.5)	88.1 (2.2)
1-NN	d_2	88.7 (0.2)	85.9 (0.1)
SVM	bigram	86.9 (0.8)	86.7 (3.0)
SVM	trigram	87.3 (0.3)	85.9 (1.7)
SLM	bigram	79.1 (0.1)	75.9 (0.4)
SLM	trigram	85.4 (0.2)	82.6 (0.3)

Table 1 shows results of experiments using three classifiers:

1. k-nearest neighbour, with $k = 1$, driven by the K-metrics (d_1 and d_2) defined by equations 1 and 2.

2. A support vector machine classifier, driven by bigram and trigram features of the sequences.

3. A Statistical Language Model, also driven by bigram and trigram features.

These techniques were applied both directly to the melody contours extracted from the MIDI files, and also to a further processed version in which the melody contour was represented by a sequence of pitch intervals rather than absolute pitches.

Table 1 summarises the results of our experiments. We note:

- The best result is achieved by using the K-metric defined by equation 1 to drive a 1-NN classifier.
- The second version of the K-metric defined by equation 2 also performed well.
- The remaining experiments, based on Statistical Language Models, generally performed less well, but as might be expected the trigram models outperformed the bigram models.
- One very clear result is that the results obtained from pitch interval melody contours are in all cases better than absolute contours. This indicates that representation plays an important role in melody classification. The same observation is also reported in [10], where authors the explain that, although the absolute pitch method can represent the original work more objectively, the interval method is more compatible with human perception, since when people memorize, distinguish or sing a melody, they usually do it based only on the interval information.

5. CONCLUSION

Our principal results are:

- A simple k-nearest neighbour classifier driven by a normalized information distance measure based on the length of a Lempel-Zip 78 parse outperforms some Statistical Language Models based on bigrams and trigrams.
- Pitch interval representation of melody seems to lead to better results than absolute pitch representation.

Of course one has to be careful in interpreting these results. We are certainly not suggesting that the best way of recognizing music genre is to employ our LZ78 based version of the K-metric on our melody contour abstraction of MIDI files, for example:

1. Sufficient careful attention to feature choice and classifier by a domain expert seems likely to outperform a universal similarity measure. For example, the melody contours we extract from the MIDI files throw away all information about note duration. Including note duration features must be a great help in recognising music genre, simply as a means of establishing its ’groove’.

2. Other compression algorithms may perform better as a basis for a K-metric. This led us to run some final experiments with the more complex LZ77 algorithm [6]: early results suggest the results are not as good as the ones reported here.

3. It would be interesting to explore more modern compression algorithms than those in the LZ families, for example the block-sorting algorithms of [19] or the PPM family [20]. The model-switching of PPM, which is clearly an advantage for compression performance, may prove something of a two-edged sword for classification purposes.

We conclude by again expressing our surprise that the universal similarity measure did so well relative to the others. Our results seem to support the conjecture of Vitanyi and his co-workers in [5] that Kolmogorov complexity based similarity metrics may have useful applications in classifications tasks.

6. REFERENCES

[1] Ponce de Len P. and Iesta J.M. ”Feature-driven recognition of music styles” *Pattern Recognition and Image Analysis: First Iberian Conference, IbPRIA 2003* Lecture Notes in Computer Science, 2652, Springer-Verlag, 2003, pp. 773-781.

[2] McKay, Cory ”Using Neural Networks for Musical Genre Classification” *http://www.music.mcgill. ca/ cmckay/projects.html* McGill University Paper, 2003.

[3] Pollastri E. and Simoncelli. G ”Classification of melodies by composer with hidden Markov models” *Proceedings First International Conference on WEB Delivering of Music* 23-24 Nov. 2001 pp. 88- 95.

[4] Pickens J. ”A comparison of language modeling and probabilistic text information retrieval approaches to monophonic music retrieval” *International Symposium on Music Information Retrieval, ISMIR 00* Plymouth, Massachusetts, October 2000.

[5] Li M., Chen X., Ma B. and Vitanyi P. ”The similarity metric” *Proc. 14th ACM-SIAM Symposium on Discrete Algorithms* 2003.

[6] Ziv J. and Lempel A. ”A Universal Algorithm for Sequential Data Compression” IEEE

Transactions on Information Theory, Vol. IT-23, No. 3, May 1977, pp. 337-343.

[7] Ziv J. and Lempel A. ”Coding Theorems for Individual Sequences” IEEE Transactions on Information Theory, Vol. IT-24, No. 4, 1977, pp. 405-412.

[8] Welch T.A. ”A Technique for High Performance Data Compression” IEEE Computer, V.17, No. 6, 1984, pp 8-19

[9] Collobert R. and Bengio S. ”SVMTorch: Support Vector Machines for Large-Scale Regression Problems” Journal of Machine Learning Research, 2001, pp 1:143-160.

[10] Chai W. and Vercoe B. ”Folk Music Classification Using Hidden Markov Models” *Proceedings of International Conference on Artificial Intelligence* June 2001.

[11] Parsons, D. ”The Directory of Tunes and musical Themes” S. Brown, Cambridge England, 1975.

[12] Barlow H. and Morgenstern S. ”A Dictionary of Musical Themes” E. Benn, London 1978.

[13] Nettl, B. ”Folk and Traditional Music of the Western Continents” 2d ed. Prentice-Hall, 1973

[14] Evans S. and Barnett B. ”Network Security Through Conservation of Complexity” *MILCOM 2002* Anaheim, CA, USA, October 7-10, 2002.

[15] Powell D.R., Dowe D.L., Allison L., Dix T.L. ”Discovering simple DNA sequences by compression” *Proceedings of Pacific Symposium on Biocomputing* 1998, pp 3:597-608.

[16] Li M. and Vitanyi P. ”An Introduction to Kolmogorov Complexity and Its Applications” Springer 1997

[17] Bennett C.H., Gacs P., Li M., Vitanyi P.M.B. and Zurek W. ”Information Distance” IEEE Transactions on Information Theory, 44:4 (1998), pp 1407-1423.

[18] Li M. and Sleep M.R. ”Improving Melody Classification by Discriminant Feature Extraction and Fusion” *Proc. International Symposium on Music Information Retrieval 2004 (ISMIR-04)* Barcelona, 11-14 Oct. 2004.

[19] Burrows M. and Wheeler D.J. ”A block-sorting lossless data compression algorithm” SRC Research Report 124, Digital Equipment Corporation, PA California, 1994.

[20] Cleary G.C. and Teahan W.J. ”Unbounded Length Contexts for PPM” The Computer Journal, V.40, No.2/3, 1997