FIRST-ORDER LOGIC CLASSIFICATION MODELS OF MUSICAL GENRES BASED ON HARMONY

Amélie Anglade
Centre for Digital Music
Queen Mary
University of London
amelie.anglade@elec.qmul.ac.uk

Rafael Ramirez
Music Technology Group
Universitat Pompeu Fabra
rramirez@iua.upf.edu

Simon Dixon
Centre for Digital Music
Queen Mary
University of London
simon.dixon@elec.qmul.ac.uk

ABSTRACT

We present an approach for the automatic extraction of transparent classification models of musical genres based on harmony. To allow for human-readable classification models we adopt a first-order logic representation of harmony and musical genres: pieces of music are represented as lists of chords and musical genres are seen as context-free definite clause grammars using subsequences of these chord lists. To induce the context-free definite clause grammars characterising the genres we use a first-order logic decision tree induction algorithm, Tilde. We test this technique on 856 Band in a Box files representing academic, jazz and popular music. We perform 2-class and 3-class classification tasks on this dataset and obtain good classification results: around 66% accuracy for the 3-class problem and between 72% and 86% accuracy for the 2-class problems. A preliminary analysis of the most common rules extracted from the decision tree models built during these experiments reveals a list of interesting and/or well-known jazz, academic and popular music harmony patterns.

1 INTRODUCTION

Users tend to be sceptical about automatic recommender systems that are not transparent. Providing some insight into the reasoning to the user has proven to improve both the user's trust and his involvement in the system [3, 11]. Thus, for a better user acceptance, automatic music classification systems (which can be used as part of a music recommender system) should provide an explanation to the user on how a piece of music is classified by the system.

Recent studies [1, 9] have shown that a logic-based representation of the musical events together with a logical inference such as Inductive Logic Programming (ILP) [5] allow for a human-readable characterisation of music. In this article we extend these works by building human readable and transparent music classification models – namely first-order logic decision tree models (an extension of the classical decision trees using ILP) – performing both classification and characterisation. We focus on classification into musical genres using as descriptor the harmony of each song.

The paper is organised as follows: In Section 2 we review some existing studies using harmony for automatic classification. In Section 3 we introduce the harmonic content description employed in this study. In Section 4 we present the details of our learning task, including the data, the inductive logic decision tree algorithm and the results (classification performances and characterisation rules obtained) before concluding in Section 5.

SMC 2009, July 23-25, Porto, Portugal

2 PREVIOUS RELATED WORK

Although some harmonic (or chord) sequences are famous for being used by a composer or in a given genre, little attention has been paid in the automatic genre recognition literature to how harmony can help in this task. For example, in [12] the authors use a chroma feature representation describing the harmonic content of the music. A comparison of the histograms reveals some patterns which contain some genre specific information. Recognition rates around 70% are reported for a five class classification. However this study focuses on low-level harmony features.

In [10], a rule-based system is used to classify sequences of chords belonging to three categories: Enya, Beatles and Chinese folk songs. A vocabulary of 60 different chords was used, including triads and seventh chords. Classification accuracy ranged from 70% to 84% using two-way classification, and the best results were obtained when trying to distinguish Chinese folk music from the other two styles, which is a reasonable result as both western styles should be closer in terms of harmony.

Paiement et al. [7] also used chord progressions to build probabilistic models. In that work, a set of 52 jazz standards was used, encoded as sequences of 4-note chords. The authors compared the generalization capabilities of a probabilistic tree model against a Hidden Markov Model

(HMM), both capturing stochastic properties of harmony in jazz, and the results suggested that chord structures are a suitable source of information to represent musical genres.

More recently, Lee [4] has proposed genre-specific Hidden Markov Models that learn chord progression characteristics for each genre. Although the ultimate goal of this work is using the genre models to improve the chord recognition rate, he also presented some results on the genre classification task. For that task a reduced set of chords (major, minor, and diminished) was used.

Finally, Perez-Sancho et al. [8] have investigated if 2, 3 and 4-grams of chords can be used for automatic genre classification on both symbolic and audio data. They report better classification results when using a richer vocabulary (seventh chords) and longer n-grams.

3 HARMONIC CONTENT DESCRIPTION AND REPRESENTATION

We extend these previous studies in which chord sequences are of fixed length by using context-free definite-clause grammars to represent chord sequences of arbitrary length.

3.1 Harmonic content

The music pieces used in this study have been kindly provided by the Pattern Recognition and Artificial Intelligence Group of the University of Alicante. There, experts have collected, annotated and double-checked files encoded in the format of the PG Music software Band in a Box (aka BIAB) [1] and then converted into MMA [2] format. These files can be seen as simplified scores only containing the chords which are labelled in a jazz/pop/rock shorthand fashion (e.g. using G7 for G dominant seventh chord, D for D major, etc.). In these transcriptions from the University of Alicante, chords are limited to major or minor triads and 7th chords (dominant seventh, major seventh or minor seventh). But there is no unique way to transcribe chords and notice that different levels of detail in chord representation might lead to the induction of different classification rules. Furthermore only the chord changes are annotated in the provided files. Although meter positions of chords are important since we do not have access to this information we leave this for future work.

3.2 Using context-free definite-clause grammars as representation scheme

Context-free definite-clause grammars proved to be useful in the logic-based extraction of biological patterns in a particular class of amino acids sequences, the neuropeptide precursor proteins (NPPs) [6]. NPPs share common characteristics with musical pieces (represented as chord sequences): these sequences are highly variable in length, they tend to show almost no overall sequence similarity and the class (NPPs or non-NPPs in the case of amino acids sequences, musical genres in the case of songs) to which a given sequence belongs is not always clear (some NPPs have not yet been discovered and experts can disagree on the genre of a given song). Both because of these similarities in the data and because context-free definite-clause grammars can be induced using Inductive Logic Programming, we choose to adopt this representation scheme.

In the definite clause grammar (DCG) formalism a sequence over a finite alphabet of letters is represented as a list of letters. Here the chords (e.g. G7, Db, BM7, F#m7, etc.) are the letters of our alphabet. A DCG is described using predicates. For each predicate `p/2` (or `p/3`) of the form `p(X,Y)` (or `p(c,X,Y)`), `X` (the input) is a list representing the sequence to analyse and `Y` (the output) is the remaining part of the list `X` when its prefix matching the predicate `p` (or property `c` of the predicate `p`) is removed.

```
%We assume the tonality is C Major
perfect_cadence(A,B):-
gap(A,C), degree(5,C,D), degree(1,D,E), gap(E,B).

%definition of the gap predicate
gap(A,A).
gap([_|A],B) :- gap(A,B).

%definition of the rootNote predicate
rootNote('C',[c|T],T).
rootNote('C',[cm|T],T).
...

%definition of the degree predicate
degree(5,A,B) :- rootNote('G',A,B).
degree(1,A,B) :- rootNote('C',A,B).
```

Table 1. Simple definite clause grammar describing a perfect cadence in C major.

To illustrate this, an example of a simple chord sequence context-free definite-clause grammar encoding the concept of perfect cadence (in C major) is given in Table 1. In this example, the target concept is `perfect_cadence/2`. To describe it three background predicates, `rootNote/3`, `degree/3` and `gap/2`, are used. `rootNote(n,A,B)` means that the first chord of list `A` has for root note `n`. `B` is the remaining list when the first chord of `A` is removed. `degree(d,A,B)` means that the first chord of list `A` has for degree `d`. The last lines of the Table 1 state that root note G corresponds to the 5th degree (dominant) in C major and C corresponds to the 1st degree (tonic). In Prolog the underscore (_) can match anything, so the `gap/2` predicate (also

[1] http://www.pgmusic.com/products_bb.htm
[2] http://www.mellowood.ca/mma/

from [6]) matches any chord sequence of any length, allowing to skip uninteresting subsequences (not characterised by the grammar rules) and to handle large sequences for which otherwise we would need very large grammars. Finally, the first lines of Table 1 define a perfect cadence as a chord on the fifth degree directly followed by a chord on the first degree (using the `degree/3` predicate), sequence that can happen anywhere in the list of chords that define the song (due to the `gap/2` predicate).

```
rootNote('C',[c|T],T).      rootNote('C',[cm|T],T).      ...
rootNote('Cs',[cs|T],T).    rootNote('Cs',[csm|T],T).    ...
...                         ...                          ...

interval(perf_uni,A,B) :- rootNote('C',A,B), rootNote('C',B,C).
interval(perf_uni,A,B) :- rootNote('Cs',A,B), rootNote('Cs',B,C).
...
interval(min_sec,A,B) :- rootNote('C',A,B), rootNote('Db',B,C).
...
interval(dim_oct,A,B) :- rootNote('C',A,B), rootNote('Cb',B,C).
...

gap(A,A).
gap([_|A],B) :- gap(A,B).
```

Table 2. Background knowledge predicates used in the first-order logic decision tree induction algorithm to describe genres. For each chord in a chord sequence its root note is identified using the `rootNote/3` predicate. The intervals between the root notes (measured upwards) are "computed" using the `interval/3` predicate.

For the genre classification tasks our target predicate is `genre/3` and the patterns we extract are based on the intervals between root notes of the chords. Root interval progressions capture some degree information but do not depend on the tonality. Thus when using root intervals no pre-processing of the data or key extraction is necessary. The background predicates used to describe our grammar (given as background knowledge to our learning system) are given in Table 2. Notice that contrary to the example in Table 1 in which one rule was enough to describe a perfect cadence, we look for a set of rules to describe each genre, each rule describing one characteristic chord sequence of this genre.

4 LEARNING CLASSIFICATION RULES FOR MUSICAL GENRES

4.1 Training data

The data set contains three genres: popular, jazz, and academic music. Popular music data consists of pop, blues, and celtic (mainly Irish jigs and reels) music; jazz consists of a pre-bop class grouping swing, early, and Broadway tunes, bop standards, and bossanovas; and academic music consists of baroque, classical and romantic music. All the categories have been defined by music experts at the University of Alicante who have also collaborated in the task of assigning meta-data tags to the files and rejecting outliers. The total amount of pieces is 856 (Academic 235; Jazz 338; Popular 283), providing around 60 hours of music data.

4.2 Learning algorithm

We have applied Tilde's top-down decision tree induction algorithm [2]. Tilde can be considered as a first order logic extension of the C4.5 decision tree algorithm: instead of testing attribute values at the nodes of the tree, Tilde tests logical predicates. This provides the advantages of both propositional decision trees (i.e. efficiency and pruning techniques) and the use of first-order logic (i.e. increased expressiveness). First-order logic enables us to use a background knowledge (which is not possible with non relational data mining algorithms). It also provides a more elegant way to represent musical concepts/events/rules which can be transmitted as they are to the users. Thus the classification process can be made transparent to the user.

Tilde builds models, namely first-order logic decision trees which can also be represented as ordered sets of rules (or Prolog programs). In the case of classification, the target predicate of each model represents the classification problem.

4.3 Learning task

We use Tilde with `genre/3` as target predicate, where `genre(g,A,B)` means the song `A` (represented as its full list of chords) belongs to genre `g`. The last argument `B`, the output list (i.e. the empty list) is necessary to comply with the context free definite clause grammar representation. The predicates considered to build the model are `interval/3` and `gap/2`, defined in the background knowledge (cf. Table 2). In addition we constrain the system to use at least two consecutive `interval` predicates between two `gap` predicates. This guarantees that we are considering local root interval sequences of a least length 2 (i.e. chord sequences of length 3) in the songs. However notice that the context free grammar definite clause representation allows these local root interval sequences to be of any length larger than 2.

4.4 Classification results

Our objective was to classify musical pieces into the three main genres present in the dataset: academic, jazz and popular music. For that we both built a model that was directly dealing with the 3-class problem and induced three models dealing with each of the 2-class subproblems. For each classification task we performed a 5-fold cross-validation.

Furthermore we controlled the complexity of the decision trees built by varying the minimal number of examples a leaf should cover (MC). The results of these experiments are shown in Table 3.

academic/jazz/popular	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.398)	0.663	**0.665**	0.655
Stderr	0.016	**0.016**	0.016
# nodes in the tree	91.0	**42.6**	26.6
# literals in the tree	248.6	**116.2**	71.0
academic/jazz	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.590)	0.839	**0.860**	0.844
Stderr	0.015	**0.015**	0.015
# nodes in the tree	22.2	**10.6**	5.2
# literals in the tree	62.6	**29.4**	14.8
academic/popular	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.540)	0.743	**0.768**	0.723
Stderr	0.019	**0.019**	0.020
# nodes in the tree	42.8	**24.0**	12.0
# literals in the tree	113.6	**61.6**	31.6
jazz/popular	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.551)	0.843	**0.819**	0.804
Stderr	0.015	**0.016**	0.016
# nodes in the tree	44.0	**21.8**	12.2
# literals in the tree	125.2	**61.0**	31.8

Table 3. Classification results (on the test data) using a 5-fold cross-validation. MC (minimal number of examples a leaf should cover) is a parameter of the decision tree learning algorithm. The number of nodes and literals present in a tree gives an estimation of its complexity.

The models for all the classification tasks have a good accuracy which is not much affected by the value of the minimal coverage of a leaf (MC): the accuracy is always much higher than the baseline. Changing the minimal coverage of a leaf from 2 examples to 5 examples (and similarly when going from MC=5 to MC=10) leads to trees containing half the number of nodes and literals (so models that are two times simpler). As long as the classification accuracy is not affected using simpler models has several advantages. Firstly, the processing time to assign a class to an unseen example is smaller when using simpler models. Moreover simpler models contain less rules and each rule covers on average a higher number of examples. Such rules tend to be more meaningful and do not capture local or isolated phenomena, so are less subject to overfitting. Finally if we want to display the model to the user (for transparency reasons) simpler models are easier to understand. Here a good compromise is reached when using MC=5 for which the classification accuracy is generally higher than for any other MC value and the models are simpler but not overly simplified.

The confusion matrices for the four classification tasks when using MC=5 are shown in Table 4. We obtain respectively 86% (academic vs. jazz), 77% (academic vs. popular), 82% (jazz vs. popular) and 67% (3-class problem) accuracy. The best results are obtained when trying to distinguish jazz from another genre (academic or popular). The biggest difficulty that appears in both the 3-class task and the 2-class task is to distinguish academic music from popular music. Indeed the harmony of these two genres can be very similar, whereas jazz music is known for its characteristic chord sequences, very different from other genres harmonic progressions.

Real/Predicted	academic	jazz	popular	Total
academic	145	31	58	234
jazz	33	267	37	337
popular	68	56	151	275
Total	246	354	246	846
academic	197	37		234
jazz	43	294		337
Total	240	331		571
academic	165		69	234
popular	49		226	275
Total	214		295	509
jazz		272	65	337
popular		46	229	275
Total		318	294	612

Table 4. Confusion matrices (on the test data) for the four classification tasks using a 5-fold cross-validation and for minimal coverage of a leaf set to 5 (MC=5).

4.5 Overview of the extracted rules

As explained in Section 4.2 for each run Tilde returns a classification model that can be represented as a tree or as an ordered set of rules (or a Prolog program). Because of space limitation we only show some interesting and recurrent rules extracted from the various models built (a complete list of classification models and their rules is available upon request). However note that a rule in itself can not perform classification both because of having a lower accuracy than the full model and because the ordering of rules in the model is important to the classification (i.e. some rule might never be used on some example because one of the preceding rules in the model covers this example).

The following rule was found in the academic vs. jazz classification models:
genre(academic,A,B) :- gap(A,C), interval(perf_fifth,C,D), interval(perf_fifth,D,E), gap(E,B).
"Some academic music pieces contain a chord root interval sequence of two consecutive ascending perfect fifth."

This rule can be interpreted as the common IV-I-V degree progression. Interestingly the same rule appears in the popular vs. jazz classification model to characterise popular music. So rather than characterising academic music or popular music this rule suggests that a sequence of two consecutive ascending fifth does not occur very frequently in jazz music (at least not as frequently as in academic or popular music).

The above rule is further specialised in the 3-class models to characterise popular music this time:
genre(popular,A,B) :- gap(A,C), interval(perf_fifth,C,D), interval(perf_fifth,D,E), interval(min_sev,E,F), gap(F,B).
"Some popular music pieces contain a chord root interval sequence of two consecutive ascending fifth directly followed by an ascending minor seventh."
This sequence could be for instance the IV-I-V-IV sequence found in the verse of *"Let it be"* by the Beatles.

Other rules found both in the academic vs. jazz and the 3-class models are:
genre(academic,A,B) :- gap(A,C), interval(min_sev,C,D), interval(perf_fifth,D,E),interval(perf_fourth,E,F),gap(F,B).
"Some academic music pieces contain a chord root interval sequence in which an ascending minor seventh is followed by an ascending perfect fifth, followed by an ascending perfect fourth."
and:
genre(academic,A,B) :- gap(A,C), interval(perf_fifth,C,D), interval(perf_fifth,D,E), gap(E,F), interval(perf_fifth,F,G), interval(perf_fourth,G,H), gap(H,B).
"Some academic music pieces contain a chord root interval sequence of two ascending perfect fifth later (but not necessarily directly) followed by an ascending perfect fifth and an ascending perfect fourth."
They can be respectively interpreted as V-IV-I-IV and IV-I-V later followed by a back and forth pattern such as I-V-I or IV-I-IV.

Some very jazzy patterns were also found, such as:
genre(jazz,A,B) :- gap(A,C), interval(perf_fourth,C,D), interval(aug_fourth,D,E), gap(E,B).
"Some jazz music pieces contain a chord root interval sequence containing an ascending perfect fourth followed by an ascending augmented fourth."
and:
genre(jazz,A,B) :- gap(A,C), interval(maj_sev,C,D), interval(perf_fourth,D,E), gap(E,B).
"Some jazz music pieces contain a root interval sequence containing an ascending major seventh directly followed by an ascending perfect fourth."

5 CONCLUSIONS AND FUTURE WORK

In this paper we presented a first-order logic approach to automatically extract genre classification models using harmony. This models are not black boxes: thanks to the expressiveness of first order logic the decision tree models we obtained can be presented to the users as sets of human readable rules. Good classification results (comparable to previous work results in the field) were obtained with these first-order decision trees algorithms. With almost no accuracy loss we managed to lower the complexity of our models from 50 rules to 25 rules on average, getting simpler, faster to use and more meaningful decision trees. By using a context-free definite-clause grammar representation which can encode chord sequences of any length we extended previous classification and characterisation studies that were limited to chords sequences of fixed length. For instance in [1] musical style was characterised using chord sequences of length 4 . In [8], the n-gram representation is used to study chord sequences of length 2, 3 or 4 only. Our system not only allows for any chord sequence length but also enables the coexistence of harmony progressions of various lengths in the same model.

Future work includes adding the chord categories (e.g. minor triad, dominant seventh, etc.) in our grammar to try to increase the classification accuracy of our models. We also plan to test if using degrees (when key estimation is possible) instead of root intervals can improve our models. Finally we intend to test our framework on audio data using chord transcription algorithms.

6 ACKNOWLEDGMENTS

This work is supported by the EPSRC project OMRAS2 (EP/ E017614/1) and partially supported by the Spanish TIN project ProSeMus (TIN2006-14932-C02-01). The first author's internship at the Music Technology Group was supported by the EPSRC Platform grant EP/E045235/1. We would like to thank the Pattern Recognition and Artificial Intelligence Group of the University of Alicante for providing the data.

7 REFERENCES

[1] A. Anglade and S. Dixon. Characterisation of harmony with inductive logic programming. In *Proceedings of the 9th International Conference on Music Information Retrieval*, pages 63–68, Philadelphia, Pennsylvania, USA, 2008.

[2] H. Blockeel, L. De Raedt, and J. Ramon. Top-down induction of clustering trees. In J. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning*, pages 53–63, Madison, Wisconsin, USA, 1998. Morgan Kaufmann.

[3] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings*

of the 2000 ACM conference on Computer supported cooperative work, pages 241–250, Philadelphia, Pennsylvania, USA, December 2000.

[4] K. Lee. A system for automatic chord transcription using genre-specific hidden markov models. In *Proceedings of the International Workshop on Adaptive Multimedia Retrieval*, pages 134–146, Paris, France, 2007.

[5] S. H. Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, 1991.

[6] S. H. Muggleton, C. H. Bryant, A. Srinivasan, A. Whittaker, S. Topp, and C. Rawlings. Are grammatical representations useful for learning from biological sequence data? - a case study. *Journal of Computational Biology*, 8(5):493–522, 2001.

[7] J.-F. Paiement, D. Eck, and S. Bengio. A probabilistic model for chord progressions. In *Proceedings of the 6th International Conference on Music Information Retrieval*, pages 312–319, London, UK, September 2005.

[8] C. Perez-Sancho, D. Rizo, S. Kersten, and R. Ramirez. Genre classification of music by tonal harmony. In *International Workshop on Machine Learning and Music, International Conference on Machine Learning*, Helsinki, Finland, July 2008.

[9] R. Ramirez, A. Hazan, E. Gómez, E. Maestre, and X. Serra. Discovering expressive transformation rules from saxophone performances. *Journal of New Music Research*, 34(4):319–330, 2005.

[10] M.-K. Shan, F.-F. Kuo, and M.-F. Chen. Music style mining and classification by melody. In *Proceedings of 2002 IEEE International Conference on Multimedia and Expo*, volume 1, pages 97–100, 2002.

[11] R. Sinha and K. Swearingen. The role of transparency in recommender systems. In *2002 Conference on Human Factors in Computing Systems*, pages 830–831, Minneapolis, Minnesota, USA, April 2002.

[12] G. Tzanetakis, A. Ermolinskiy, and P. Cook. Pitch histograms in audio and symbolic music information retrieval. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, pages 31–38, Paris, France, 2002.