Automatic Polyphonic Music Composition Using the EMILE and ABL Grammar Inductors*

David Ortega-Pacheco and Hiram Calvo

Centro de Investigación en Computación, Instituto Politécnico Nacional,
Av. Juan de Dios Bátiz s/n, esq. Av. Mendizábal, México, D. F., 07738. México
dortegab06@sagitario.cic.ipn.mx, hcalvo@cic.ipn.mx
www.hiramcalvo.com

Abstract. In this paper we present a method for automatic polyphonic music composition using the ABL and EMILE grammar inductors. To evaluate the performance of the EMILE and ABL engines we use a voting classification scheme based on TF-IDF weighting—we show a novel adaptation of the n-gram concept for music classification. We performed experiments with six musical MIDI collections from a different classical music composer each (Bach, Chopin, Liszt, Schubert, Mozart and Haydn). For each composer we applied our method to obtain five new polyphonic music compositions, and then we tested the membership of the new compositions with regard to every composer. We found that new compositions have similar membership to the set of composer styles than natural compositions. We conclude that our method is capable of creating new, relatively original compositions in the same musical style of each author.

Keywords: Grammar Induction, Automatic Music Composition, EMILE Grammar Inductor, ABL Grammar Inductor, M-Grams, TF-IDF weighting.

1 Introduction

The goal of grammar induction (or grammar inference) is to learn in a supervised or unsupervised way the syntax of a particular language from a corpus of this language. Grammar induction algorithms have been used in several areas, for example Computational Linguistics, Natural Language Processing, Bio-Informatics, Time Series Analysis, Computer Music, etc. Particularly in Computer Music, unsupervised grammar induction algorithms such as ECGI, K-TSI and ALERGIA have been used for automatic music composition [2]. In this work, we explore the possibility of using other unsupervised algorithms for grammar induction in automatic music composition. We used the EMILE (Entity Modeling Intelligent Learning Engine) and ABL (Alignment-Based Learning) grammar inductors; EMILE and ABL have been successfully used in Natural Language Processing tasks.

The ABL engine is based on sequence analysis and induces the structure (obtaining a grammar) by aligning and comparing each input sequence [7–9], see Fig. 1. The

---

* We thank the support of Mexican Government (SNI, SIP-IPN, COFAA-IPN, and PIFI-IPN).

Fig. 1. The ABL Algorithm

Fig. 2. The EMILE Algorithm

EMILE algorithm is based on categorical grammars and attempts to learn the grammatical structure of a language from positive sentences of that language [3, 10], see Fig 2.

Our previous work in [5] shows the use of EMILE for automatic monophonic music composition. In [5], we show an approach to obtain a numerical representation of monophonic musical MIDI files (Musical Instrument Digital Interface) which in turn constitute a musical corpus; in that work we propose a simple method to evaluate the EMILE performance measuring the intersection of single notes of the new compositions and the musical corpus. In this work, we continue such research, proposing a methodology for automatic music composition using EMILE and now ABL too. We include now the capability of handling polyphonic MIDI files [6]. In addition we propose a more robust evaluation scheme.

The methodology presented here roughly consist of the following steps: first we obtain a numerical representation of polyphonic MIDI files to generate a musical corpus based on the encoding proposed in [5], see Section 2.1; then we use the ABL or the EMILE grammar inductors to find the grammar of the musical corpus, see Section 2.2. Finally, the obtained grammar is used to create new musical compositions, see Section 2.3.

Automatic composition lacks of a standard defined method of evaluation; this task is especially risky in the sense that grading a new automatic musical composition can be very subjective, and the difference of what is music and what is not music is not always clear. Therefore, for the evaluation of our system, we propose a new scheme which consists of a voting classification, based on the TF-IDF weighting—previously used successfully in document classification [1, 4]. This voting classification scheme relies in a new adaptation of the n -gram concept that consists in determining a fixed value of n (window size) according to the musical concept of time signature, instead of the number of notes. With the proposed *musical n -grams*, the number of notes that fit in a window is variable (See section 3).

2 Methodology for Automatic Music Composition

The block diagram of the system for automatic music composition is shown in Fig. 3. The following sections describe in detail the stages of *musical corpus*, *grammar induction* and *music composition*.

Fig. 3. Block diagram for automatic music composition

2.1 Musical Corpus

The musical corpus consists of musical bars extracted from each musical MIDI file in the training set. Each text line in the musical corpus contains the notes that form a musical bar and a vector of features represents each note into the bar. To be able to process polyphonic MIDI files we merge the overlapping notes present in each bar of each track to obtain one track with all the information per bar. See Fig. 4

Fig. 4. Building of the Musical Corpus

2.2 Grammar Induction

The grammar induction process takes as input the musical corpus obtained in the previous stage. Once the grammar induction on the musical corpus is done by ABL and EMILE, the result consists of two hierarchical structures of music composition rules (the first by ABL and the second by EMILE), see Fig. 5.

Fig. 5. Grammar Induction by ABL and EMILE

2.3 Music Composition

The music composition stage is based on the bar generator algorithm (Fig. 6). This algorithm takes the grammar rules obtained by ABL or EMILE grammar inductors to generate new bars. The algorithm for bar generation consists of selecting a random derivation from the first production rule $\{0\}$, and then continues deriving randomly until a string of only-terminals is reached. Note that the output of ABL is converted from its own output format to grammar rules like those produced by EMILE (Fig. 5).

Fig. 6. Music composition process

3 Evaluation Scheme

In order to evaluate the performance of our automatic music composition system, we use a voting classification scheme based on TF-IDF (Term Frequency – Inverse Document Frequency) weighting. We expect that the new automatically obtained compositions have similar membership to the set of composer styles than natural compositions.

To confirm this, we will compare three different sets of musical compositions using the voting classification scheme: (1) The musical corpus of author X, MC_X ; (2) the five new musical compositions for that author, NC_X obtained with our system; and (3) Five original compositions of the same author X, not present in the original MC_X , namely OC_X .

We build a confusion matrix consisting of the similarity of MC_A with OC_B for each author A and each author B in the collection, yielding a vector of similarities (each

row of MC_A). We call this vector MC_A-OC_B . Then we build another matrix consisting of the similarity of MC_A and the new compositions NC_B . We obtain again a vector of similarities for each author A in the collection (each row of MC_A). We call this vector MC_A-NC_B for the author A.

At this point, we have two similarity vectors for each author X, $MC-OC_X$ and $MC-NC_X$. We expect that these two vectors are closely similar to prove that new compositions belong to the same characteristic style than the original compositions.

In addition, the voting classification scheme that we propose uses a new adaptation of the n-gram concept based on the time signature. This is explained below.

3.1 Musical Concept of N-Gram

The conventional method for obtaining n-grams from a sequence of symbols consists of defining a fixed value of n symbols (window size)—see Fig. 7A. Usually, the best value of n is determined experimentally. In the domain of music, it is possible to extract n-grams in two ways: (a) fixed number of notes and variable time window; and (b) variable number of notes and fixed time window. (a) uses n as the number of contiguous notes, the time window is variable. (b) is based on musical bars. Musical bars are musical sequences with certain parameters previously defined, such as bar measure. This measure represents a natural segmentation of a bar and each segment can be considered as an n-gram, a *musical n-gram* (Fig. 7B). A musical n-gram, or for simplicity **m-gram**, contains a variable number of notes, as opposed to the traditional n-gram where the number of notes is pre-defined by the n number. Here the time window is fixed. Based on (b) we extract the m-grams as elements of the voting classification scheme for each generated music composition and each musical corpus.

Fig. 7. Musical m-grams (n-grams in the bar = Number of Frames n)

3.2 M-Grams for TF-IDF Weighting

Let C be a musical composition to be classified, and $FC_{C,mg}$ represents the frequency of the musical m-gram mg in the musical composition C . To compute the weight of a m-gram mg for a given musical corpus MC , we define $F_{MC,mg}$ as the number of times

that the m-gram mg occurs along the musical corpus MC , divided by the total number of m-grams present in the musical corpus MC . $N_{A,mg}$ is defined as the number of times that the m-gram mg occurs in any training corpus divided by the total number of m-grams across any training corpus. Then, we can define the term frequency TF of an m-gram mg in a musical corpus MC as follows:

$$TF_{MC,mg} = \frac{F_{MC,mg}}{N_{A,mg}} \quad (1)$$

We define the document frequency DF_{mg} as the number of musical corpus in which the m-gram mg occurs at least once divided by the total number of musical corpus. In our experiments, DF ranges from 0 to 6, as we are using 6 different Musical Corpus. Then, the Inverse Documents Frequency is defined as follows:

$$IDF_{mg} = \frac{1}{DF_{mg}^2} \quad (2)$$

The associated weight W for a m-gram mg to a musical corpus MC is defined as follows:

$$W_{MC,mg} = TF_{MC,mg} \cdot IDF_{mg} \quad (3)$$

Finally, using the TF-IDF weight of each m-gram we can compute the similarity measure $SIM_{MC,C}$ for a given composition C with a given musical corpus MC using eq. (4).

$$SIM_{MC,C} = \frac{\sum_{mg \in C} [FC_{C,mg} \cdot W_{MC,mg}]}{\sum_{mg \in C} FC_{C,mg}} \quad (4)$$

4 Experiments and Results

For the experimentation, we chose six classical music composers: Bach, Chopin, Liszt, Haydn, Mozart and Schubert. We chose musical MIDI files for each composer with the same time signature. Other MIDI files with different time signatures were excluded from this experiment. The parameters chosen were: time signature: 4/2; frames per bar: 4; base mark: 1/2; mark length: 480 ticks. For each composer we collected a corpus of 2000 bars for training (MC) and finally we carried out 2 experiments, described below:

4.1 Experiment 1 – Reference

We chose fifteen original compositions (OC) from each composer (this compositions were not included in a musical corpus) and use the voting classification scheme to

determine the similarity of an original composition with its corresponding composer corpus (OC_X vs MC_X). We expect that the similarity for an original composition is higher when is compared with his composer, and lower with other composers, as it can be seen in Table 1.

Table 1. OC versus MC

	Schubert	Bach	Liszt	Mozart	Haydn	Chopin
Schubert	1	0.0941	0.0774	0.1469	0.1905	0.2315
Bach	0.0501	1	0.4077	0.0808	0.8168	0.1339
Liszt	0.2339	0.2986	1	0.1021	0.3712	0.5200
Mozart	0.9007	0.3897	0.5959	0.9405	1	0.3623
Haydn	0.0091	0.0027	0.0010	0.0016	1	0.0022
Chopin	0.6662	0.2238	0.0833	0.0971	0.1520	1

4.2 Experiment 2 – Automatic Composition Performance

In this experiment, using the method of automatic music composition described in Section 0, we create five melodies for each composer and for each grammar inductor. Then, we used the voting classification scheme described in Section 0 to determine the similarity of each composition obtained with each composer. This allows us to determine the performance of our method automatic music composition using theEMILE grammar inductor and using the ABL grammar inductor. We expect each row to have similar results with respect to the Experiment 1; for example, that the vector (row) of Chopin is similar to the vector (row) of Chopin in Table 1.

Table 2. NC versus MC using EMILE (NC-EMILE)

	Schubert	Bach	Liszt	Mozart	Haydn	Chopin
Schubert	1	0.0002	0.0001	0.0003	0.0002	0.0003
Bach	0	1	0.0002	0.0001	0.0001	0.0002
Liszt	0.0002	0.0001	1	0	0	0
Mozart	0.0002	0.0001	0	1	0.0003	0.0003
Haydn	0.0001	0.0001	0	0.0002	1	0.0002
Chopin	0.0003	0.0005	0.0001	0.0002	0.0002	1

Table 3. NC versus MC using ABL (NC-ABL)

	Schubert	Bach	Liszt	Mozart	Haydn	Chopin
Schubert	1	0.0087	0.0022	0.0042	0.0153	0.0108
Bach	0.0006	1	0.0066	0.0013	0.0029	0.0028
Liszt	0.0005	0.0031	1	0	0.0006	0.0001
Mozart	0.0129	0.0456	0.0253	1	0.0156	0.0093
Haydn	0.0089	0.0018	0.0002	0	1	0.0004
Chopin	0.0038	0.0066	0.0036	0.0015	0.0048	1

Table 4. Average of absolute differences of each vector (row) of NC-EMILE (Table 2) with each vector (row) of OC (Table 1) and each vector (row) of NC-ABL (Table 3) with each vector (row) of OC (Table 1)

	OC-NC-EMILE	OC-NC-ABL
Schubert	0.8768	0.8835
Bach	0.7519	0.7542
Liszt	0.7458	0.7464
Mozart	0.4488	0.4667
Haydn	0.9974	0.9992
Chopin	0.7965	0.7996
Average	0.7695	0.7749

Table 4 is calculated by averaging the absolute differences of each vector in NC-EMILE with each vector in OC, and correspondingly for NC-ABL. For example, to calculate the performance of the composition system using ABL for the composer Chopin, the operations are:

$$\frac{|0.6662 - 0.0038| + |0.2238 - 0.0066| + |0.0833 - 0.0036| + |0.0971 - 0.0015| + |0.1520 - 0.0048| + |1 - 1|}{6} = 0.7996$$

5 Conclusions and Future Work

We can see from the results of our experiments, that there is more confusion between the original compositions and the original corpus (MC-OC_X, Table 1) for each author X compared with the new compositions and the original corpus (MC-NC_X, Table 2 and Table 3) (See Section 0 for a description of MC-OC_X and MC-NC_X). This has several interpretations; first, that the newly created compositions belong, effectively, to each author, and it is unlikely that they are confused with other authors' compositions—this is a good effect; second, that the newly created compositions are not as *fresh* as original compositions (OC), because they lose certain degree of *confusion* with other authors. In broad terms, we expected this to happen: the combination of inferred rules from MC is limited, whereas a new creation NC can have expressions not seen in MC.

Note, in Table 4, however, that when comparing the relative set of similarities per author, that the system reproduces adequately the behavior of the original compositions. Particularly for the case of Haydn, the performance is practically identical. In average, both induction engines, EMILE and ABL yield similar results. EMILE is slightly superior over ABL for all composers.

With regard to the evaluation method, it is important to mention that the musical m-grams we suggest provide a quantifiable number that helps to determine what is happening after the grammar induction. It grades the quality of the inferred rules: a very high similarity between MC and NC means that the system is not changing things very much, NC should be similar, but not *very similar* to MC. NC should be similar to MC as OC to MC.

We established a new framework for evaluation that can be used by similar systems. Also we explored the possibility of using grammar inductors for the musical

language, making a parallelism between music and human language. The results of this work can be further explored by trying more variants resembling those applied in computational linguistics, such as the concept of *musical synonyms*.

References

1. Black, J.A., Ranjan, N.: Automated Event Extraction from Email. Final Report of CS224N/Ling237 course in Stanford (2004)
2. Alcázar, C., Pedro, P., Ruiz, E.V.: A Study of Grammatical Inference Algorithms in Automatic Music Composition and Musical Style Recognition. In: Workshop on Automata Induction, Grammatical Inference, and Language Acquisition. The Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, TN, USA (1997)
3. Eric, D.: Extension of the EMILE algorithm for inductive learning of context-free grammars for natural languages. Master's Thesis, University of Dortmund, UK (1997)
4. Manning Christopher, D., Hinrich, S.: Foundations of Statistical Natural Language Processing, pp. 541–544. MIT Press, England (2000) (second printing)
5. David, O.-P., Calvo, H.: Music Composition using the EMILE Grammar Inductor. In: Gelbukh, A., Kuri, A. (eds.) Advances in Artificial Intelligence and Applications, Research in Computing Science, pp. 341–351 (2007)
6. Selfridge-Field, Eleanor: Beyond MIDI, the Handbook of Musical Codes, pp. 41–72. The MIT Press, Cambridge (1997)
7. van Zaanen, M.: ABL: Alignment-Based Learning. In: Proceedings of the 18th International Conference on Computational Linguistics COLING, pp. 961–967 (2000)
8. van Zaanen, M.: Bootstrapping structure using similarity. In: Monachesi, P. (ed.) Computational Linguistics in the Netherlands, pp. 235–245 (1999)
9. van Zaanen, M.: Bootstrapping syntax and recursion using alignment-based learning. In: Procs. 17th International Conference on Machine Learning ICML, pp. 1063–1070 (2000)
10. Vervoorte, M.: Emile 4.1.7 User Guide. University of Amsterdam (2004)