A Machine-Learning Approach to Autonomous Music Composition

Ryan N. Lichtenwalter, Katerina Lichtenwalter and Nitesh V. Chawla

The University of Notre Dame, Notre Dame, IN 46556, U.S.A.

ABSTRACT: Several music composition systems exist that generate blues chord progressions, jazz improvisation, or classical pieces. Such systems often work by applying a set of rules explicitly provided to the system to determine what sequence of output values is appropriate. Others use pattern recognition and generation techniques such as Markov models. These systems often suffer from mediocre performance and limited generality. We propose a system that goes from raw musical data to feature vector representation to classification models. We employ sliding window sequential machine learning techniques to generate classifiers that correspond to a training set of musical data. Our approach has the advantage of greater generality than explicitly specified musical grammar rules and the potential to apply a wide variety of powerful existing non-sequential learning algorithms. We present the design and implementation of the composition system. We demonstrate the efficacy of the method, show and analyze successful samples of its output, and discuss ways in which it might be improved.

KEYWORDS: music, composition, sequential learning, feature extraction, sliding window, data mining

1. INTRODUCTION

This paper represents a cross-disciplinary approach for leveraging techniques germane to the domain of machine learning to accomplish significant feats in the domain of fully autonomous music composition. Successful knowledge discovery in

Correspondence: Ryan N. Lichtenwalter, Interdisciplinary Center for Network Science and Applications, The University of Notre Dame, Notre Dame, IN 46556, U.S.A.
e-mail: rlichten@nd.edu; klichten@nd.edu; nchawla@nd.edu

a large sample of existing compositions such that software can learn how to write a musically appealing composition is at once inspiring and challenging. The problem is that music is a high-dimensional space with varying degrees of interdependency: durations, pitches, vertical sonorities, dynamics, articulations, etc. Further, its structure does not immediately suggest a single most efficacious approach to modeling. We demonstrate how sliding windows can be used to generate a powerful model, which can then be applied in a recurrent sliding window to compose musically interesting and relevant compositions.

The great novelty of this work, and its difference from other autonomous composition systems, is in the feature extraction and sliding window ideas. Although conceptually simple, together they form a powerful general framework for handling music composition in a way different from any methods of which we are aware. The feature set can be extended in any number of ways, the approach can be extended to handle less fundamental musical concepts such as articulations and dynamics, and an incredible variety of existing high-performance non-sequential learning algorithms may be applied.

The general approach consists of three phases as visualized in Fig. 1: data pre-processing for converting to a feature vector representation suited to machine learning, data analysis including rule inference, and nondeterministic music composition. The main theoretical challenge faced by this approach lies in the second and third steps: to produce rules that are both sufficiently general and

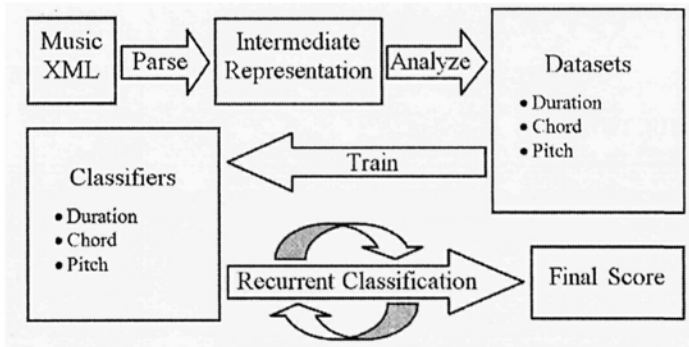


Fig. 1: The complete composition process from MusicXML on which to train to an output score.

relevant to the nature of tonal music, and also to make the best use of these rules to produce musical works that qualitatively resemble tonal music as described by the training set. We present the musical source data and our method of transforming it to a form from which standard non-sequential classifiers can learn.

2.0 DATA SET CONSTRUCTIO N

We present the musical source data, some properties of that data, and our method of transforming it to a "learnable form" by several standard classifiers.

2.1 Source Data

We limited our analysis and predictions to Western tonal music using exclusively Bach chorales as training data. Tonal music is based upon the principles of well-defined durations and pitches. The availability of a large collection of Bach chorales online in various formats made them a natural choice. The exclusive use of Bach chorales confined the generality of the compositional task greatly. We further limited the selection of training data to four-part chorales without instrumental accompaniment; each part is monophonic. Although MIDI is an established standard in digital music transmission, it fails to convey many useful musical characteristics and parsing it to reconstruct the full musical data available in a score is cumbersome, error-prone, and incomplete. We thus chose to use MusicXML as source data. The intention of the MusicXML format is to preserve musical information as it would appear on a page of printed music, with the complete information necessary to perform a piece or apply music-theoretic analysis to it. In total, we trained on 157 chorales comprising 41,414 duration and pitch data set instances, and 9,856 chord data set instances.

Performing analysis directly on transfer formats such as MusicXML is both difficult and inefficient. We created our own simple object model, which better serves the needs of the various scanning and analysis components of data set generation over the musical score. The model not only allows for a lossless representation of duration, chord, and pitch information, but also provides a contextualized structure in which that information can be successfully analyzed.

2.2 Refactoring Tonalities

Although we recognize that there may be fundamental differences in the chord progressions and common intervals between differing modes, we assume that within a given mode, the tonalities may all be reduced to a single representative. Using MusicXML key designations, we transpose all major key pieces to C major and all minor key pieces to A minor. By doing the transposition, we obtain data sets of larger size. One trivial artifact that emerged in the process of transposition is that the pitch ranges for the various parts do not always correspond to the real pitch ranges in the Bach chorales. In the future, more sophisticated transpositions could prevent this.

Noteworthy is that the division between the pieces designated major and the pieces designated minor is not strictly correct or meaningful. Frequent modulations render these designations arbitrary. It may therefore make sense in future work to either remove the artificial separation into major and minor data sets, or to divide instances among data sets based on Schenkerian key reckonings. Despite the modal confusion that occurs in the segregated major and minor data sets, we found that there is a difference in their predictability.

2.3 Sliding Window Data sets

We limited the set of features to those we could construct with basic duration and pitch information and higher level properties of the containing measures, parts, and scores. We produced 8 pitch datasets: one for each of the soprano, alto, tenor, and bass parts in both major and minor key. We produced 12 duration datasets: one for each part in 4/4, 3/4, and 3/2 time. We produced 2 chord datasets: one for major key and one for minor key. In general, the framework allows for automatic construction of data sets based on available training data and composition requests supplied to the system. We now illustrate the process by which we created data sets. The reference example is visible in Fig. 2.

In music, proximate preceding durations largely dictate the valid set of potential future durations. Sliding windows can capture this relationship by specifying that a duration has as features all those durations preceding it within the window. Table 1 illustrates the features in the sliding window. The absolute temporal location of an event within a measure and the position of the measure within the piece also influence its duration. These location features, which are not sequential in nature, are also included in the dataset.
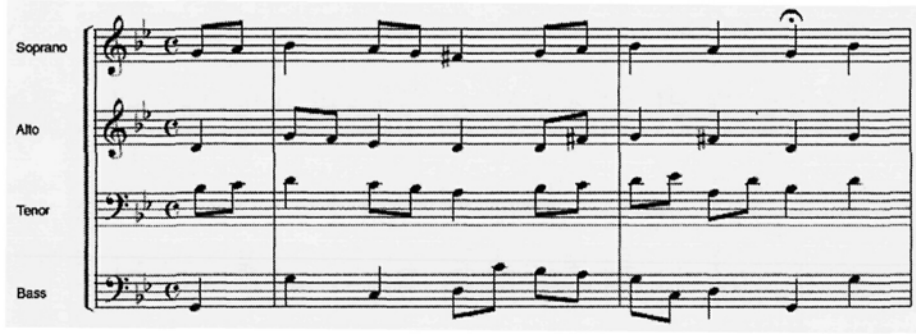


Fig. 2: Sample music from which the examples are drawn.

We divided the source data according to differing time signatures and then produced separate data sets for each time signature. The need for such data segmentation is clear. Scores with 3/2 time signature, for instance, could lead to rules predicting a whole note on the second beat. If such rules were applied on the second beat in a 4/4 measure, the result would be illegal output. Including the time signature as a feature can help produce rules contraindicating such illegal predictions, but it is safer to simply divide the training data. Aside from blatantly illegal output, there are also stylistic differences on various beats that depend upon the time signature that may otherwise be very difficult to learn. Likewise, because of stylistic differences across a selection of parts, we also segmented duration data sets according to part. Although durations and duration sequences are relatively homogeneous across four-part chorales, this does not apply in orchestral scores with instruments of very different capabilities. Consider training a duration classifier on a piccolo part and using it to produce a tuba part.

Pitches in polyphonic music are applicable to individual notes, but they are also influenced by and important in defining the interplay of multiple notes that are simultaneously audible. We can thus say that in addition to their own pitches, musical events have as a feature the chords of which they are members. Generating sequences of chords defines tonal polyphonic music, and thus it is necessary to create data sets and classifiers to handle music generation in the broad polyphonic context.

An ideal method, automatic Schenkerian analysis, remains an open problem in the field of computer music. Schenkerian analysis is the process by which chord names may be determined based on a sophisticated, somewhat subjective inter-dependent consideration of vertical and horizontal sonority. Some of the

TABLE 1.

Table illustrating the Construction of duration data sets[1]

Position		Standard Features		Sliding Window Duration Features					Class
Measure	Beat	Meas. Type	Tick	n-5	n-4	n-3	n-2	n-1	n
0	4	pickup_first	576	-	-	-	-	-	eighth
0	4.5	pickup_first	672	-	-	-	-	eighth	eighth
1	1	middle	0	-	-	-	eighth	eighth	quarter
1	2	middle	192	-	-	eighth	eighth	quarter	eighth
1	2.5	middle	288	-	eighth	eighth	quarter	eighth	eighth
1	3	middle	384	Eighth	eighth	quarter	eighth	eighth	quarter
1	4	middle	576	eighth	quarter	eighth	eighth	quarter	eighth
1	4.5	middle	672	quarter	eighth	eighth	quarter	eighth	eighth
2	1	middle	0	eighth	eighth	quarter	eighth	eighth	quarter
2	2	middle	192	eighth	quarter	eighth	eighth	quarter	quarter
2	3	middle	384	quarter	eighth	eighth	quarter	quarter	quarter

[1]Note: The values in this table are based on the example in Figure 1. This is based on the soprano line in the example.

difficulties with performing such analyses are explained well by Marsden (Marsden 2007). Marsden, Smoliar, and others have proposed methods to facilitate Schenkerian analysis and perform chord naming with results comparable to human analysis (Marsden 2007, Smoliar 1999); but, to our knowledge nobody has claimed to achieve successful fully autonomous chord analysis using purely computational methods.

Because we choose not to attempt automated analysis and manual chord analysis is intractable, we instead rely on a surrogate for chord information as a predictor of both upcoming chords and current pitches. This surrogate is the set of unique pitches, disregarding registral information, to compose a particular chord. Table 2 and Fig. 3 demonstrate how the feature labels are constructed. The method is not musically correct, but it provides an approximation with significantly less complexity.

TABLE 2

The translation system whereby a particular pitch may be encoded as a unique prime number equivalent

Pitch	E^b	B^b	F	C	G	D	A
Fifth Degree	-3	-2	-1	0	1	2	3
Prime Index	5	3	1	0	2	4	6
Prime	13	7	3	2	5	11	17

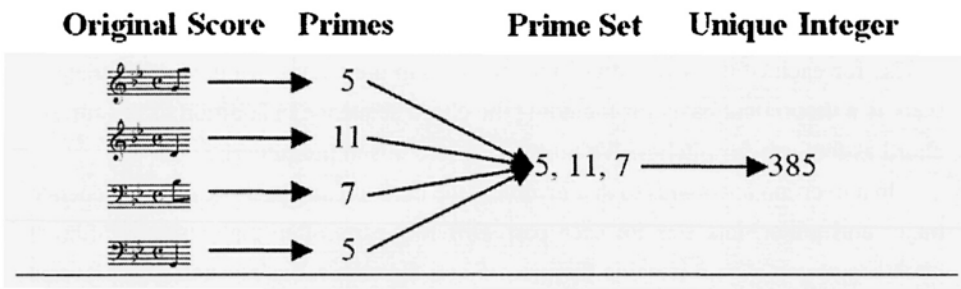


Fig. 3: The manner by which a set of pitches across parts is translated into a single consistent integer representing the vertical sonority.

Table 3 illustrates how the chord data set would look using Schenkerian analysis. The table shows the appearance of the chord data sets actually used in training and generation. Some relationships between the integers and chord names are apparent after study in Table 4. The discrepancies in integer-to-name matches are the result of passing tones, which Schenkerian analysis disregards, but which are nevertheless present in the vertical sonorities. The vertical sonorities create a longer sequence of noisier features due to passing tones, as is evident in the comparative lengths of Table 3 and Table 4.

Although the raw vertical sonorities may present good features for pitch prediction, more problematic is the effect of using vertical sonorities in generating sequences of chords. A theoretically correct analysis of Bach yields a large number of cadences and other well-established sequences indicative of rules. Merely using vertical sonorities fails to disregard passing tones not strictly useful in determining the real chord sequence, thus fragmenting the set of supporting instances for the classifier. The accuracy analysis in Section 4.1 suggests that there is a lot of room for improvement if we could improve the analysis of chords or obtain data sets with chord names specified.

In a sliding window context there are two primary sequential features for pitch. The first is the sequence of pitches in a particular part, which directly influence the upcoming pitch by dictating the interval necessary to reach the pitch. The second is the sequence of chords in a particular part, which will influence the upcoming pitch indirectly by first influencing the upcoming chord. Since we first predicted an upcoming chord and then predicted upcoming pitches within all the parts, the sequence of previous chords can be replaced by the chord within which the pitch to predict will fit. The current chord has a much more direct influence on pitch predictions, and experiments showed that the current chord feature has an information gain of 1.97 as compared to the much weaker information gain of 0.22 or less for each of the sequential chord features in the sliding window. Nonetheless, there is a theoretical basis for including the chord sequence in addition to the current chord as features for pitches. We hope to explore this in the future.

In a decision analogous to that involving the duration data sets, we created separate major and minor data sets for each part. Different parts often play entirely different pitch sequences. Again imagine the piccolo and the tuba. This decision also prevents parts from leaving their ranges when a pitch overlaps the range of another part.

TABLE 3

Table illustrating the construction of chord data sets assuming chord names are available[1]

Position		Standard Features		Sliding Window Chord Features					Class
Measure	Beat	Meas. Type	Duration	n-5	n-4	n-3	n-2	n-1	n
0	4	pickup_first	quarter	-	-	-	-	-	i
1	1	middle	quarter	-	-	-	-	i	i
1	2	middle	quarter	-	-	-	i	i	ii^{o6}
1	3	middle	eighth	-	-	i	i	ii^{o6}	V
1	3.5	middle	eighth	-	i	i	ii^{o6}	V	V^7
1	4	middle	eighth	i	i	ii^{o6}	V	V^7	i^6
1	4.5	middle	eighth	i	ii^{o6}	V	V^7	i^6	vii^{o6}
2	1	middle	eighth	ii^{o6}	V	V^7	i^6	vii^{o6}	i
2	1.5	middle	eighth	V	V^7	i^6	vii^{o6}	i	iv^7
2	2	middle	quarter	V^7	i^6	vii^{o6}	i	iv^7	V
2	3	middle	quarter	i^6	vii^{o6}	i	iv^7	V	i

[1]Note: The values in this table are based on the example in Fig. 1. This is based on the bass line in the example.

TABLE 4

Table illustrating the actual construction of chord data sets[1]

Position		Standard Features		Sliding Window Chord Features					Class
Measure	Beat	Meas. Type	Duration	n-5	n-4	n-3	n-2	n-1	n
0	4	pickup_first	eighth	-	-	-	-	-	385
0	4.5	pickup_first	eighth	-	-	-	-	385	1870
1	1	middle	eighth	-	-	-	385	1870	385
1	1.5	middle	eighth	-	-	385	1870	385	1155
1	2	middle	eighth	-	385	1870	385	1155	442
1	2.5	middle	eighth	385	1870	385	1155	442	910
1	3	middle	eighth	1870	385	1155	442	910	7667
1	3.5	middle	eighth	385	1155	442	910	7667	15334
1	4	middle	eighth	1155	442	910	7667	15334	385
1	4.5	middle	eighth	442	910	7667	15334	385	1394
2	1	middle	eighth	910	7667	15334	385	1394	385
2	1.5	middle	eighth	7667	15334	385	1394	385	910
2	2	middle	quarter	15334	385	1394	385	910	7667
2	3	middle	half	385	1394	385	910	7667	385

[1]Note: The values in this table are based on the example in Fig. 1. This is based on the bass line in the example.

TABLE 5

Table illustrating the construction of pitch data sets[1]

Location		Standard Features			Sliding Window Pitch Features					Class
Measure	Beat	Meas. Type	Duration	Chord	n-5	n-4	n-3	n-2	n-1	n
0	4	pickup_first	quarter	385	-	-	-	-	-	D
1	1	middle	eighth	385	-	-	-	-	D	G
1	1.5	middle	eighth	1155	-	-	-	D	G	F
1	2	middle	quarter	442	-	-	D	G	F	E^b
1	3	middle	quarter	7667	-	D	G	F	E^b	D
1	4	middle	eighth	385	D	G	F	E^b	D	D
1	4.5	middle	eighth	1394	G	F	E^b	D	D	$F^\#$
2	1	middle	quarter	385	F	E^b	D	D	$F^\#$	G
2	2	middle	quarter	7667	E^b	D	D	$F^\#$	G	$F^\#$
2	3	middle	quarter	385	D	D	$F^\#$	G	$F^\#$	D
2	4	middle	quarter	385	D	$F^\#$	G	$F^\#$	D	G

[1]Note: The values in this table are based on the example in Fig. 1. This is based on the alto line in the example.

The complete set of features for the pitch data sets is visible in Table 5 based on the alto part of the excerpt. The chord feature for a given pitch instance is always the chord at the time the pitch is sounded regardless of other chords that may occur while the pitch is sustained, a possible weakness in the current implementation.

Finally, although pitch and duration are linked (Friberg 1991), we thought it necessary to allow the duration of a pitch at time n to be a feature of the pitch data set, since a combined (pitch, duration) class would be both computationally intensive and insufficiently general. For instance, the raised fifth scale degree occurring on a triplet figure is so specific that without a much larger data set it would be difficult to derive meaningful rules using such a combined class. Further, duration is a feature of the pitch data set, rather than the other way around, because the composition order predicts the duration class first during composition. Our decision to predict the duration of an event before its pitch was mostly arbitrary, but was also an effort to determine the most restrictive information first.

3.0 MUSICAL CLASSIFICATION

Music is partially related to and determined by broad static features. These might include the name of a part or instrument, the time signature, the tempo, and the type of measure. Many of these serve as non-sequential features in our data sets either by segregating the data into distinct sets or by appearing a single time each instance to provide guidance to classifiers. Such information is related to the object-property traits of music. A score has parts, each of which has measures, each of which has additional divisions and descriptive information. Observing these objects can help in training classifiers, but the power of such features is limited. Most of the defining characteristics of music are dynamic: sequences of different durations, chords, and pitches.

In general, training classifiers on the non-sequential features can be handled by a large body of supervised non-sequential learning algorithms. Our goal was to create a generalized framework to provide classifiers with information encoded in the standard non-sequential format that would still carry the power of the dynamic sequential features of music. This information, which appears in the data set instances as a window of events, captures a given portion of a musical duration, chord, or pitch sequence at each time unit.

We sought to use off-the-shelf classifiers to predict characteristics of a yet uncreated musical event based on these characteristics. The goal output, however, is not a single classification but the generation of a score, an entire series of generative classification tasks. This recursive classification is one in which the results from the previous output serve as additional input in the updated window for upcoming classification. Because a blank score begins with no information whatsoever, it is necessary to seed it with the basic elements required by each of the sliding window classifiers. The seeding may be replaced by the random generation of a set of first notes based on training data.

After seeding a blank score with values sufficient to define an initial instance in the duration, chord, and pitch data sets for the event, it is possible to apply any non-sequential classifier. The output from this classifier is used to define a new instance in which feature window now includes the previously predicted instance.

Because a generated musical event may consist of a single note, multiple notes, or no notes at all in the form of rests across all parts, it may have associated with it the same set of characteristics that garnered individual data sets in Section 0: duration, pitch, and chord. To generate a sensible musical event, all these classes have to be created appropriately and in a reasonable order. To this end, it is fortunately possible to consider many characteristics of musical events separately. In some cases, such as with duration and pitch, there is little dependence between characteristics, which is substantiated later in Table 8 by the minimal decrease in entropy produced by the duration feature in predicting pitch. As an example, previous pitches and chords dictate new pitches and chords, but they do not account for upcoming sequences of durations. Likewise, sequences of durations dictate appropriate upcoming sequences of durations, but they do not as firmly dictate pitches.

Generating polyrhythmic scores that also fit into a generated sequence of chords is significantly more difficult than generating homorhythmic scores. The former task can proceed in one of two ways. First, a chord can be generated with a given duration and then individual parts can be constructed in the context of the chord. Second, some subset of the parts can have pitches generated that help define the chord, and then remaining parts can be filled. Both suffer from the difficulty that generating independent durations fitting into the encompassing chord duration is too restrictive, and allowing durations to extend beyond the duration of the encompassing chord enables the inadvertent modification of upcoming chords. Essentially, it is difficult to create a score in which chords transition due to one part at a time changing pitch to

cause the transition while still maintaining the integrity of the chords. We highlight two main high-level approaches from which one may choose in applying the compositional framework.

Polyrhythmic Approach (without chords):

```
for part in score do
    for measure in part do
        beat = 0
        while beat < measure.beats do
            duration = PREDICT-DURATION(durationFeatures)
            pitch = PREDICT-PITCH(pitchFeatures,duration)
            beat += duration.beats
        endwhile
    endfor
endfor
```

Homorhythmic Approach (with chords):

```
for measure in score do
    beat = 0
    while beat < measure.beats do
        duration = PREDICT-DURATION(durationFeatures)
        chord = PREDICT-CHORD(chordFeatures)
        for part in score do
            pitch = PREDICT-PITCH(pitchFeatures,duration,chord)
        endfor
        beat += duration.beats
    endwhile
endfor
```

The first approach allows for rhythmic variation across parts, but makes it difficult to incorporate chords in a consistent and reasonable manner because pitches can change out from beneath the chords in which they are supposed to fit. The second allows for consistent chord consideration, but prevents autonomy within the parts. The second approach allows for music that is tonally coherent even if it is

rhythmically less interesting, thus we generated scores using the second approach, but the successful marriage of the two approaches is a necessary condition for producing output that is truly qualitatively similar to Bach chorales.

3.1 Learning Algorithms

Many of the characteristics of music are naturally suited to rule-based or decision-based algorithms. Consider the case of designing the rhythm to fit into a measure in common time (4/4). Suppose that the first three beats are quarter notes and the final beat is an eighth note. This accounts for 87.5% of the measure. In no cases will any of the training data ever support durations greater than 12.5% of the measure because such an instance would not complete a valid measure. It must also be true that no instances in the training data will support a note value of less than 12.5% unless there is an accompanying instance in the training data that accounts for the remainder of the measure. If there were no such accompanying instance, the measure would be incomplete.

Placed in musical terms, if an eighth note is selected to complete the measure, 100% of the measure is accounted for. If a sixteenth note is selected, it means that there must be at least one instance in the training data supporting such a prediction. Such an instance, however, will necessarily support the completion of the measure with either a series of note durations of lesser value or a final sixteenth note.

In support of these statements and the ability of the algorithms to learn, here are two of the specific rules that they produced. This is output from the Ripper algorithm. Other rule-based and decision-based classifiers produced similar rules.

Rule 1: **if** n_tick $\geq$ 672 (7/8 measure) **then** n_duration = eighth
Rule 2: **if** n−1_duration = dotted-quarter **then** n_duration = eighth

The second rule corroborates the musically intuitive supposition that dotted quarter notes are frequently followed by eighth notes. In fact, the rule statistics illustrate that in the Bach chorales on which we trained, every dotted quarter note is followed by an eight note. The existence and subsequent automatic derivation of these types of rules is precisely what drives the accurate and valid duration component of the output compositions.

For pitches, rules are not as easily rendered because they tend to be much less strict. Unlike with durations for which illegal instances are clear, rules regarding pitch are less so. Still, especially given the backing of the n_chord feature, there is direct information about what set of pitches is likely to complete the specified vertical sonority. For a C major chord, $C^{\#}$, or its enharmonic counterpart, D^{b}, is detrimental to the consonance of the chord to the point of being illegal. An even more obvious case is when the chord feature indicates a rest through all parts of the score. If there is such a rest, then each of the parts must also have a rest. Rules along these lines are illustrated below.

Rule 3: **if** n_chord = 1023 **then** 4:B:0
Rule 4: **if** n_chord = 1 (no pitches) **then** rest

The n_chord feature equals the multiplicative identity only when there are no pitches in the vertical sonority to donate their primes as factors. In the cases when n_chord is 1023, the diminished triad composed of B, D, and F is the underlying chord. In this case, for the soprano part in the data sets based on minor chorales the pitch B4 is predicted with high confidence.

3.2 Randomization and Non-determinism

While the model must encompass the rules dictated by the musical source data, it should also avoid repetitive sequences. Consider the simplified toy example of two rules DE→D and ED→E and an input of DE. Although a reasonably sophisticated model will make such sequences highly improbable, they may occasionally occur. The solution is to introduce non-determinism. The model still defines the rules that guide generation of the composition, but strong rules that can cause repetitive sequences are prevented from doing so. We intersect a uniformly distributed random variable with the cumulative distribution function of the class prediction probabilities. The class under the point of intersection is the predicted class. This ensures that class values that are most musically appropriate have the highest probability of occurring, but that any prediction supported to some degree by training data is possible. The complexity of the chord and pitch models made this unnecessary, but the duration model was dominated by the quarter note class and required application of this technique. Fig. 4 illustrates the concept.

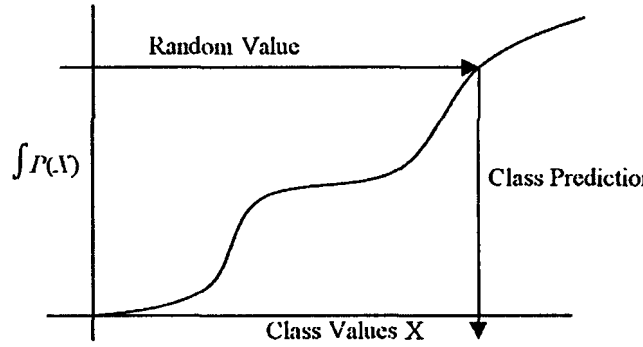


Fig. 4: A method for introducing non-deterministic behavior into the classification output for greater diversity.

4. RESULTS

We provide theoretical support for the efficacy of the approach and a brief musical analysis of actual output samples.

4.1 Performance Metrics

Before examining compositional output, it is interesting and instructive to relate basic learning metrics based on our translation from music to features and classes. For these metrics, predictions have not been randomized. Because the recurrent sliding window approach is used, each classification represents both the class output of a window at position k and input for sequential feature $n-1$ at position $k+1$. Therefore, classification error is theoretically propagated. Unlike in many other sequential classification tasks, however, we believe such propagation causes little damage. So long as each prediction is logical in the context of the previous predictions, the error at any single point is relatively unimportant. The need for substantiating training data will mostly prevent musically bizarre sequences. For this reason, we currently have no results related to the way in which error is propagated throughout the predictions.

Table 6 illustrates the results of performing information gain analysis on the features we selected for determining duration. The beat of the note to predict is by far

TABLE 6

Information gain analysis for the duration data sets

Feature	InfoGain
n Beat	0.51
n-1 Duration	0.25
n-2 Duration	0.06
Measure Type	0.05
n-3 Duration	0.04
n-4 Duration	0.03
n-5 Duration	0.03

the most important attribute. This corresponds to the fact that, especially toward the end of a measure, the location at which an event occurs is heavily constrained by the remaining time that may be legally allocated. The feature with the second highest information gain is the duration of the note before the note to predict. Dotted durations are often followed by a duration that fills the fractional portion of time. For instance, dotted quarter notes are followed by eighth notes and dotted half notes are followed by quarter notes. It is likely that the previous duration feature is a powerful predictor in other instances too, but these are less obvious. Finally, the remaining durations appear with an order of magnitude less information gain and their usefulness as predictors is questionable at best. We expected measure type to be an important feature because of its ability to discriminate the beat disparity in pickup measures and their counterparts at the end of pieces, but the data above show that this is not the case.

Table 7 represents information gain across both the minor and major chord data sets. The information gain for all features in the minor data sets was slightly higher than for the major data set features. This may suggest that pieces designated with minor keys follow a more rigid or predictable set of rules. The information gain for the previous chord is reasonable, and information gain for all of the chordal features is much higher than for the temporal features. This suggests that the appearance of a particular chord is generally independent of measure position and position in the piece. Of course, this is musically incorrect. Many rules exist that tie chords to their locations in the score in the Bach chorales. Just to name a few, the last measure must

contain the tonic chord, the penultimate measure most frequently contains the dominant chord, the cadential six-four chord, may only occur on a strong beat unless it precedes a half cadence and the tonic typically occurs on strong beats. That considerations of temporal locale have so little effect in reducing entropy may suggest a high degree of confusion in the appearance of particular chord types on strong or weak beats. Most certainly, the information gain is also mitigated by the usage of vertical sonority names rather than true chord analysis results.

TABLE 7

Information gain analysis for the chord data sets.

Feature	InfoGain
n-1 Chord	1.97
n-2 Chord	1.36
n-3 Chord	1.12
n-4 Chord	1.03
n-5 Chord	0.97
n Duration	0.28
Measure Type	0.14

TABLE 8

Information gain analysis for the pitch data sets.

Feature	InfoGain
n Chord	1.83
n-1 Pitch	1.36
n-2 Pitch	0.80
n-3 Pitch	0.59
n-4 Pitch	0.48
n-5 Pitch	0.42
n Duration	0.08
Measure Type	0.04

The information gain analysis for the pitch training set in Table 8 is also telling. By far the most significant entropy-reducing feature is the previous pitch. In tonal music, the set of intervals that are acceptable at a given instance is highly constrained within the context of the 88 pitches available on the piano. The specific musical source data, Bach chorales, probably boosted the value of the information gain. Bach chorales are further constrained to vocal ranges, which are significantly smaller than instrumental ranges. Further, the size of common melodic intervals in the training data was no more than an octave.

Every pitch behind the previous pitch has a successively lower information gain. While they appear to be somewhat better predictors for pitch than duration, pitches *n-2* and beyond are significantly less important than pitch *n-1*.

Fig. 5 illustrates the accuracy achieved by several common classifiers implemented in WEKA (Witten and Eibe 2005). Specifically, the figure represents the performance of naive Bayes, C4.5 (Quinlan 1993), Ripper (Cohen 1995), and a nearest neighbor classification algorithm (Martin 1995) across each of the three classification tasks required by the automated composition system. We obtained all accuracies by performing 10 variably seeded runs of 10-fold cross validation.

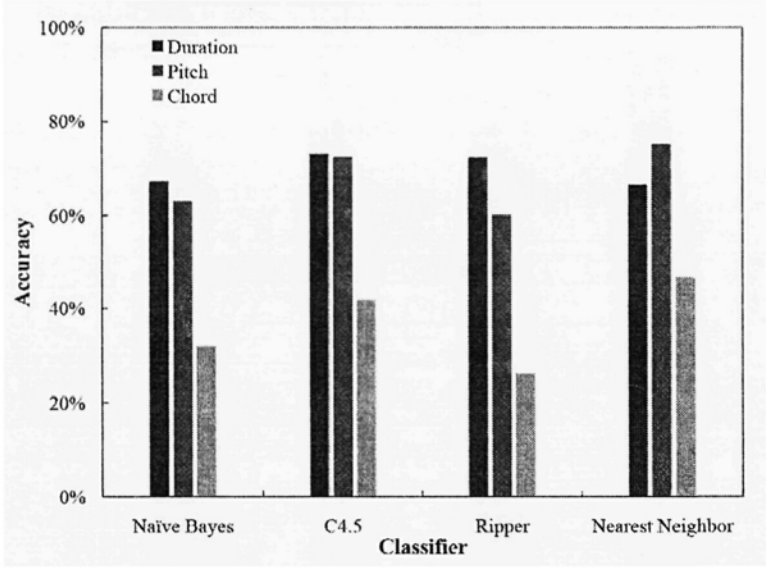


Fig. 5: The 10-fold cross validation accuracy of four families of classifiers on the duration, pitch, and chord data sets.

Immediately evident is that the various musical classification tasks, and the data sets that support them, have different properties. The ease of the classification tasks varies greatly. The duration task, the first of the sources of input in the recurrent classifications, shows the highest accuracies across all classifiers. Fig. 5 includes both 3/4 and 4/4 data sets, but not the 3/2 data set due to data sparseness. In the 4/4 data sets, classification was generally easier, with a top mean accuracy across parts of 75.8% by C4.5. The 4/4 soprano data set yielded the highest accuracy: 80.7%.

The chord classifiers achieved much lower accuracies than their duration and pitch counterparts. There are several explanations for the low accuracy, many of which have already been partially provided. Because of the rudimentary nature of chord reckoning, the chord data sets are all very noisy. Consider that there are only a few dozen separate chord names under Schenkerian analysis that one could reasonably expect in the Bach chorales; yet, we observe hundreds of distinct vertical sonorities. In the chord data set containing the pieces designated as minor, the chord data set with the larger range of possible chord values, there were 236 distinct vertical sonorities. This number disregards inversions and pitch doubling while the few dozen distinct chord names even include inversions. The much smaller range of class values visible to a classifier operating on real chords should automatically improve accuracy just because its chances are higher. More musically important, however, is the set of rules that real chord progressions follow, which the noisy vertical sonorities follow less often. We have little doubt that the installation of a good chord analysis model into the system could drastically improve chord classification performance.

Also true is that many of the vertical sonorities predicted by the classifier in testing may have been musically valid in the context of the sequential features provided. Further analysis is necessary to determine to what extent the incorrect classifications were actually invalid classifications. To such an end, it would be helpful to define a metric capable of indicating, beyond mere accuracy, the musical appropriateness of a particular model based on some of the output it provides. For the classification task, it would then be ideal to maximize this appropriateness metric in preference to accuracy.

Finally, the pitch data set achieved accuracies as high as 75% over all major and minor data sets. The best highest accuracy was 74.1% accuracy on the minor data set and 76.3% accuracy on the major data set. Overall, classification of new instances with the minor data set was somewhat more difficult with 1%-2% lower performance for all classifiers.

An interesting and curious result is the difference in classifier performance across the part data sets. Classification accuracy proceeds from highest to lowest in the order bass part, soprano part, alto part, tenor part. All classifiers for both the major and minor data sets exhibited this characteristic with no exceptions. In some cases, the differences were extreme. The bass part classifications were 5%-10% more accurate than the inner voice classifications. The soprano classifications were also much more accurate than the inner voice classifications. This is somewhat surprising because the bass part has the largest range of the four, while the inner voices have comparatively restricted ranges. This would seem at least superficially to make classification more difficult because a wider range of values exists for the class. The bass and soprano parts do enjoy some rules that do not hold for the inner voices. Soprano intervals are theoretically more predictable because they must be melodic. There are fairly stringent rules about skips and leaps. They should be few in number and they should be followed by step movement in the opposite direction. The bass must land on the root of the tonic chord in the last measure. Still, the inner voices are expected to move relatively little, which could serve to restrict the effective range of reasonable class values.

4.2 Compositional Output

The compositions were generated by the classifiers using data sets with the largest number of instances. The time signature of all pieces presented in this section is 4/4, because there were only about 700 instances in each of the 3/4 duration data sets. The 4/4 duration data sets had an order of magnitude more instances and, probably as a result, 6% higher accuracy. Both minor key and major key pitch and chord data sets were used to generate one piece each. The major key chord and pitch classifiers achieved higher accuracies than the minor key classifiers. Overall, however, we judge the output of the minor key compositions to be more aesthetically appealing. Fig. 6, 7, and 8 are samples of output. We judge the work by chord structure and functionality, dissonance and leading-tone resolution, voice doubling, voice range, intervallic progressions, cadences, and overall logic of phrasing.

The sample composition in Fig. 6 is an example of the output produced by using classifiers from the set of pieces indicated to be in major key. The first chord is provided to the system as a seed for the recurrent sliding window process. Chord construction and chord sequences exhibit a wide range of acceptability with several
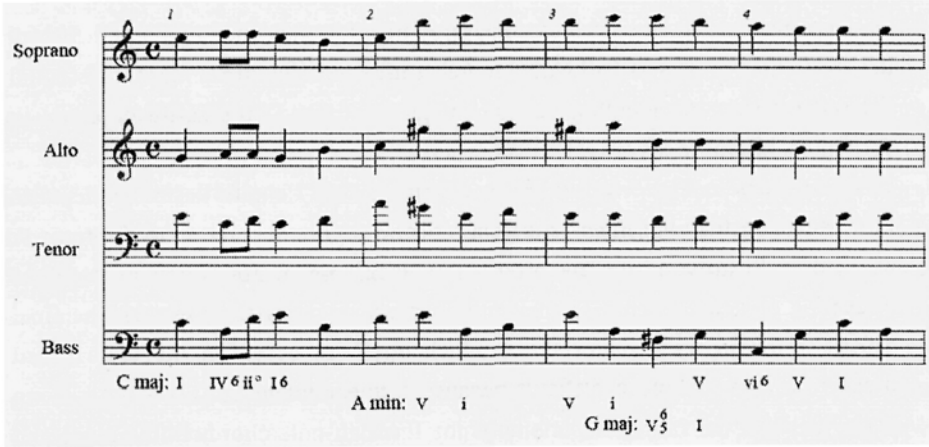


Fig. 6: The first four measures of compositional output created using the 4/4 time signature, major key data set.

progressions of chords being very nearly, but not quite correct due to excessive doubling or missing chord components. In these four measures and later in the piece the ranges of some voices tend to go too high, an artifact of indiscriminate registral shifting during training set transpositions. The melodic intervals in the soprano part create a coherent and appealing line of music with frequent change of direction and only one leap, followed as expected by step. The texture is contrapuntal and features a lot of desirable contrary motion. Phrasing suffers from rhythmic monotony and a lack of strong cadence at the end of this four-bar segment.

Two of these problems, excessive pitch doubling and rhythmic monotony are easily explained. The pitch doubling results from the fact that each of the parts has its pitch predicted independently of the others. Therefore, even if the chord progression is correct and the vertical sonority that all of the pitch predictors receive is viable, they may all independently choose to select the same pitch in the sonority, leaving out other necessary pitches. The sequential pitch features make it unlikely for all four parts to select the same pitch based on the chord information, but there are no rules to prevent the parts from all containing the same pitch. This condition is easily rectified by modifying the information that the pitch predictors receive about the chord with the pitch predictions of other parts.

The rhythmic monotony is the result of the excessively strong showing of quarter notes in the chorales, a form of skew in the training set. Often, rhythms of

interest appeared after the first few measures of output as rarer sequential feature values, once they appeared, motivated more exotic predictions. Standard differential sampling methods of handling class imbalance caused a significant drop in performance, but more investigation is needed.

The excerpts in **Fig. 7** and 8 are in A minor. **Fig. 7** shows desirable stepwise motion in reasonable sonorities. In **Fig. 8**, there is a good resolution to the tonic A chord with a Picardy third. The progression lacks proper resolution of both the leading tone and the seventh of the implied dominant chord. However, the final cadence reinforced by a major third, which does not occur on the tonic chord anywhere else in the piece, is a clear assimilation of Bach's style. This feature, along with the fact that the last chord is one of the few half-note chords in the piece and that the measure concludes with a rest, suggests that specification of measure type is useful for both the duration and pitch classifiers. The classifiers recognized and learned from Bach's typical cadential progression in the last measure of chorales in minor keys.

Analysis of the two generated pieces allows us to draw several conclusions about the rules learned by the program. These conclusions are drawn not exclusively from the excerpts above but from our entire collection of generated pieces. The rules most likely learned by the classifiers are: the tonic chord should be almost always preceded by the dominant; the leading tone must be raised in secondary dominant chords; the leading tone must be resolved upward by step; excessive skips are to be avoided in
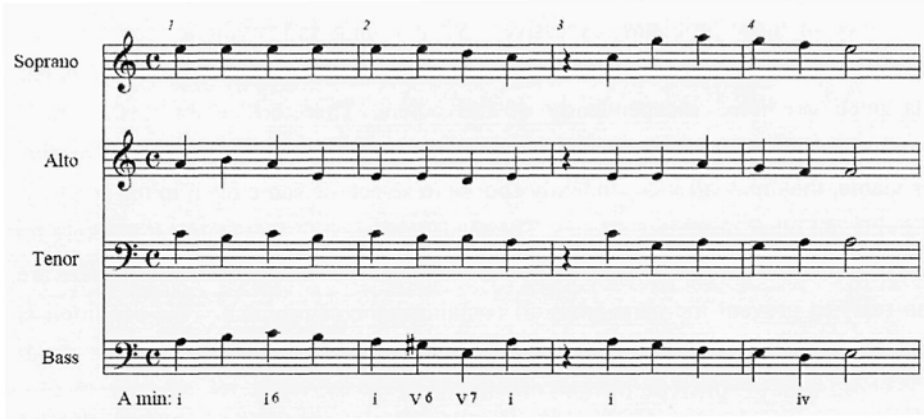


Fig. 7: The first four measures of compositional output created using the 4/4 time signature, minor key data set.

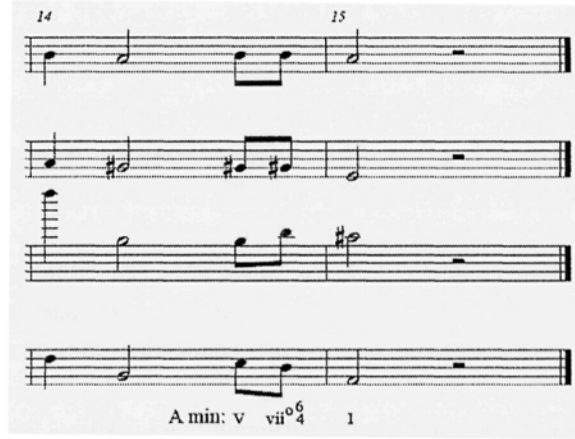


Fig. 8: The last two measures of compositional output created using the 4/4 time signature, minor key data set.

all but the bass voice; and duration sequences must fit precisely into the measures containing them. They most likely did not learn the proper resolution and doubling rules as witnessed by unresolved sevenths of the chord and the rules of standard harmonic progression; there were many incomplete and non-functional chords. In the areas where the chord and pitch classifiers were most deficient, using proper chord names would solve almost all problems.

5. CONCLUSION

Researchers have dedicated a great deal of work to improving the efficacy of automated composition. Approaches have ranged from directly specifying rules to using sequential learning techniques such as Markov chains and results have served both casual research interests and commercial products. This work is different in its application of recurrent sliding windows to musical generation. This allows for the various characteristics of music, such as rhythm, pitch, chords, and others to each use a standard non-sequential learning algorithm with the optimal inductive bias.

Despite the power of the technique, there are also several difficulties. Perhaps the most severe and persistent of these is that, as it is implemented here, it is limited to creating unconnected sequences of chords and pitches. It has no knowledge of broader score concerns such as phrasing and repetition. The severely limited way in which chords are recognized and handled also greatly mitigates the accuracy of

chord classifiers and the realism of the chords and chord progressions that are composed.

Nonetheless, the results are encouraging. The compositional output compares very favorably to recent work in the field. After laying the tremendous groundwork necessary to implement a musical object model, create an extensible mapping from musical compositions to sliding window feature sets and classes, perform musical analysis, and interface digitally with the MusicXML representation, extending the system is easy. It is possible to generate music that looks and sounds reasonable based purely on rules discovered by the application of relatively suboptimal classifiers and rudimentary chord handling. Early successive improvements will undoubtedly dramatically improve output quality and realism. In short, this paper serves as an introduction into a new way to approach the application of machine learning and data mining to the domains of musical understanding, recognition, and generation. The approach provided here enables generalized handling of the compositional process and allows the application of a wide variety of classifiers for each of the broad, intersecting dimensions of music.

6. FUTURE WORK

Our work presents several opportunities for enhancements and breakthroughs. Most obviously, the compositional system cannot currently create polyrhythmic scores; the task remains to find the optimal method to handle difficult musical interdependencies between chords, durations, and pitches of the parts that compose them. Another potential improvement lies in sliding window direction. Forward sliding window approaches are not always the most efficacious. One example of this is illustrated by Dietterich in the context of a letter processing task (Dietterich 2002). In music, it is likely that the same applies. The recurrent sliding window seed should have the highest stability possible with respect to the rest of the piece and this may be better achieved by starting on the final chord rather than on the first chord. Working backward also allows for applying correct enharmonic spellings in stepwise pitch sequences. It may be efficacious to consider pitch differently in the work, opting for a less musically faithful consideration of pitches that collapses an infinite set of enharmonic spellings into a single true pitch. Along these same lines, an alternative handling of pitch might instead disregard absolute pitch altogether and

use intervals (Conklin 2002). Such an approach has the advantage of an even more generalized capture of the underlying musical information, but it can also complicate the simultaneous consideration of other important factors such as part ranges and chordal information. Aside from the direct extensions of the system described here, there are numerous areas of related research where the approach could be helpful. As an example, automated Schenkerian analysis may itself benefit from sliding window techniques.

7. RELATED WORK

In (Cope 2004), David Cope discusses his recent music composition tool, entitled Gradus. It reads data in the form of short musical exercises and derives a small set of rules. Using backtracking and recursion, Gradus is able to compose simple contrapuntal pieces, but it does not initiate composition on its own: a user must explicitly supply the correct *cantus firmus* variable. The CHORAL project (Ebcioglu 1988) uses traditional chorale composition algorithms to harmonize existing chorales. It works by interpolating inner voices when given an existing bass and soprano line. Impro-Visor similarly requires a figured bass and improvises a melody atop it using probabilistic grammar modeling (Keller & Morrison 2007). Work by Smoliar in (Smoliar 1999) is dated, but may elucidate the form a generalized framework for autonomous Schenkerian analysis could take. Friberg's paper aims for a human to define rules and a framework for automated composition (Friberg 1991). Once the challenge of fully automated composition is sufficiently addressed, Friberg's rules could be used for dynamic predictions based on sets of durations. Bresin and Friberg (2000) further addressed qualitative aspects of music in terms of post-processing on autonomously constructed compositions.

Several researchers have applied rule-learning approaches to music generation in the past. A review of the application of machine learning methods for musical modeling and generation appears in (Dubnov et al. 2003). Specifically, there have been other applications of sequential learning algorithms. Laine and Kuuskankare applied genetic algorithms to generate short musical sequences in (Laine & Kuuskankare 1994). Others have used Markov chains to recognize musical patterns and then apply those patterns to automatically generate music (Verbeurgt et al. 2004). These and other approaches apply sequential learning techniques to the

composition task, but none of which we are aware achieve the flexibility of the recurrent sliding window framework provided here.

ACKNOWLEDGMENTS

Thanks to Margaret Greentree, who manages the jsbchorales.net website, for the MusicXML data.

REFERENCES

Bresin, R., Friberg, A. 2000. Emotional coloring of computer-controlled music performances. *Computer Music Journal* 24(4) 44-63.

Cohen, W.W. 1995. Fast effective rule induction, in: *The Twelfth International Conference on Machine Learning*, Tahoe City, California, USA, Morgan Kaufmann 115-123.

Conklin, D. 2002. Representation and discovery of vertical patterns in music. In: *The Second International Conference on Music and Artificial Intelligence*, Springer-Verlag 32-42.

Cope, D. 2004. A musical learning algorithm. *Computer Music Journal* 28(3) 12-27.

Dietterich, T.G. 2002. Machine learning for sequential data: A review. In: *The Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, Springer-Verlag 15-30.

Dubnov, S., Assayag, G., Lartillot, O., Bejerano, G. 2003. Using machine-learning methods for musical style modeling. *Computer Magazine* 36(10) 73-80.

Ebcioglu, K. 1988. An expert system for harmonizing four-part chorales. *Computer Music Journal* 12(3) 43-51.

Friberg, A. 1991. Generative rules for music performance: A formal description of a rule system. *Computer Music Journal* 15 56-71.

Holmes, G., Donkin, A., Witten, I.H. 1994. Weka: A machine learning workbench, in: *The Second Australia and New Zealand Conference on Intelligent Information Systems*, Brisbane, Australia 357-361.

Keller, R., Morrison, D. 2007. A grammatical approach to automatic improvisation, in: *The Fourth Sound and Music Conference*, Lefkada, Greece.

Laine, P., Kuuskankare, M. 1994. Genetic algorithms in musical style oriented generation. In: *The First IEEE Conference on Evolutionary Computation.* Volume 2. 858-862.

Marsden, A. 2007. Automatic derivation of musical structure: A tool for research on schenkerian analysis, in: *The 8th International Conference on Music Information Retrieval*, Vienna, Austria, Austrian Computer Society 19-34.

Martin, B. 1995. Instance-based learning: Nearest neighbor with generalization. Master's thesis, University of Waikato, Hamilton, New Zealand.

Quinlan, R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, California, Morgan Kaufmann.

Smoliar, S.W. 1999. A computer aid for schenkerian analysis. *Computer Music Journal* 4(2) 41-59.

Verbeurgt, K., Dinolfo, M., Fayer, M. 2004. Extracting patterns in music for composition via Markov chains. *Innovations in Applied Artificial Intelligence* 1123-1132.

Witten, I. H and Eibe F. 2005. *Data mining: practical machine learning tools and techniques*. San Francisco, California, USA, Morgan Kaufmann.