CHORD RECOGNITION IN SYMBOLIC MUSIC USING SEMI-MARKOV CONDITIONAL RANDOM FIELDS

Kristen Masada
School of EECS
Ohio University, Athens, OH
km942412@ohio.edu

Razvan Bunescu
School of EECS
Ohio University, Athens, OH
bunescu@ohio.edu

ABSTRACT

Chord recognition is a fundamental task in the harmonic analysis of tonal music, in which music is processed into a sequence of segments such that the notes in each segment are consistent with a corresponding chord label. We propose a machine learning model for chord recognition that uses semi-Markov Conditional Random Fields (semi-CRFs) to perform a joint segmentation and labeling of symbolic music. One benefit of using a semi-Markov model is that it enables the utilization of segment-level features, such as segment purity and chord coverage, that capture the extent to which the events in an entire segment of music are compatible with a candidate chord label. Correspondingly, we develop a rich set of segment-level features for a semi-CRF model that also incorporates the likelihood of a large number of chord-to-chord transitions. Evaluations on a dataset of Bach chorales and a corpus of theme and variations for piano by Beethoven and Mozart show that the proposed semi-CRF model outperforms a discriminatively trained Hidden Markov Model (HMM) that does sequential labeling of sounding events, thus demonstrating the suitability of semi-Markov models for joint segmentation and labeling of music.

1. INTRODUCTION AND MOTIVATION

Harmonic analysis is an important step towards creating high level representations of tonal music. High level structural relationships form an essential component of music analysis, whose aim is to achieve a deep understanding of how music works. At its most basic level, harmonic analysis requires the partitioning of music into segments along the time dimension, such that the notes in each segment correspond to a musical chord. This *chord recognition* task can often be time consuming and cognitively demanding, hence the utility of computer-based implementations. Reflecting historical trends in artificial intelligence, automatic approaches to harmonic analysis have evolved from purely grammar-based and rule-based systems [6, 15], to systems employing weighted rules and optimization algorithms [8, 11, 13, 14], to data driven approaches based on supervised machine learning (ML) [9, 10]. Due to their requirements for annotated data, ML approaches have also led to the development of music analysis datasets containing a large number of manually annotated harmonic structures, such as the 60 Bach Chorales introduced in [9], and the 27 theme and variations of TAVERN [2].

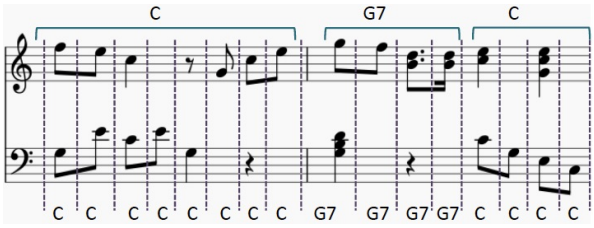

Figure 1. Segment-based recognition (top) vs. event-based recognition (bottom), on measures 11 and 12 from Beethoven WoO68.

A relatively common strategy in ML approaches to chord recognition is to break the musical input into a sequence of short duration spans and then train sequence tagging algorithms such as HMMs to assign a chord label to each span in the sequence (at the bottom in Figure 1). The spans can result from quantization using a fixed musical period such as half a measure [10] or constructed from consecutive note onsets and offsets [9]. Variable-length chord segments are then created by joining consecutive spans labeled with the same chord symbol (at the top in Figure 1). A significant drawback of these short-span tagging approaches is that segments are not known during training and inference, therefore the ML model cannot use features that capture properties of segments that are known to be relevant with respect to their harmonic content. The chordal analysis system of Pardo and Birmingham [8] is an example where the assignment of chords to segments takes into account segment-based features, however it uses a processing pipeline where segmentation is done independently of the subsequent chord labeling.

In this paper, we propose a machine learning approach to chord recognition in which a semi-Markov CRF model is trained to do joint segmentation and labeling of symbolic music. Also called segmental CRFs, this class of models can exploit features that look at all the notes in-

side a segment. Correspondingly, we define a rich set of features that capture the extent to which the events in an entire segment of music are compatible with a candidate chord label. When evaluated on a dataset of Bach chorales, the semi-CRF approach obtains a 15% error reduction over an event-tagging system. Substantially larger improvements in event-level and segment-level performance are observed on a more difficult corpus of theme and variations by Beethoven and Mozart, thus validating empirically the modeling advantage of joint segmentation and labeling.

2. SEMI-MARKOV CRF MODEL FOR CHORD RECOGNITION

Since harmonic changes may occur only when notes begin or end, we first create a sorted list of all the note onsets and offsets in the input music, i.e. the list of *partition points* [8]. A basic music *event* [9] is then defined as the set of notes sounding in the time interval between two consecutive partition points. Let $\mathbf{s} = \langle s_1, s_2, ..., s_K \rangle$ denote a segmentation of the musical input $\mathbf{x}$, where a segment $s_k = \langle s_k.f, s_k.l \rangle$ is identified by the indices $s_k.f$ and $s_k.l$ of its first and last events, respectively.

Let $\mathbf{y} = \langle y_1, y_2, ..., y_K \rangle$ be the vector of chord labels corresponding to the segmentation $\mathbf{s}$. The set of labels can range from coarse grained labels that indicate only the chord root [14] to fine grained labels that capture mode, inversions, added and missing notes [3], and even chord function [2]. Here we follow the middle ground proposed by Radicioni and Esposito [9] and define a set of labels that encode the chord root (12 pitch classes), the mode (major, minor, diminished), and the added note (none, fourth, sixth, seventh), for a total of 144 different labels. Since the labels do not encode for function, the model does not require knowing the key in which the input was written.

A semi-Markov CRF [12] defines a probability distribution over segmentations and their labels as shown in Equations 1 and 2. Here, the global segmentation feature vector $\mathbf{F}$ decomposes as a sum of local segment feature vectors $\mathbf{f}(s_k, y_k, y_{k-1}, \mathbf{x})$, with label y_0 set to a constant "no chord" value. The ensuing factorization of the distribution enables an efficient computation of the most likely segmentation $\arg\max_{\mathbf{s},\mathbf{y}} P(\mathbf{s}, \mathbf{y}|\mathbf{x}, \mathbf{w})$ using a semi-Markov analogue of the Viterbi algorithm [12].

$$P(\mathbf{s}, \mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{e^{\mathbf{w}^T \mathbf{F}(\mathbf{s},\mathbf{y},\mathbf{x})}}{Z(\mathbf{x})} \quad (1)$$

$$Z(\mathbf{x}) = \sum_{\mathbf{s},\mathbf{y}} e^{\mathbf{w}^T \mathbf{F}(\mathbf{s},\mathbf{y},\mathbf{x})}$$

$$\mathbf{F}(\mathbf{s}, \mathbf{y}, \mathbf{x}) = \sum_{k=1}^{K} \mathbf{f}(s_k, y_k, y_{k-1}, \mathbf{x}) \quad (2)$$

Following Muis and Lu [7], for faster inference, we further restrict the local segment features to two types: *segment-label features* $\mathbf{f}(s_k, y_k, \mathbf{x})$ that depend on the segment and its label, and label *transition features* $\mathbf{g}(y_k, y_{k-1}, \mathbf{x})$ that depend on the labels of the current and previous segments. The corresponding probability distribution over segmentations is shown in Equations 3 to 5 below.

Given an arbitrary segment s and a label y, the vector of segment-label features can be written as $\mathbf{f}(s, y, \mathbf{x}) = [f_1(s, y), ..., f_{|\mathbf{f}|}(s, y)]$, where the input $\mathbf{x}$ is left implicit in order to compress the notation. Similarly, given arbitrary labels y and y', the vector of label transition features can be written as $\mathbf{g}(y, y', \mathbf{x}) = [g_1(y, y'), ..., g_{|\mathbf{g}|}(y, y')]$. In Section 3 we describe the set of segment-label features $f_i(s, y)$ and label transition features $g_j(y, y')$ that are used in our semi-CRF chord recognition system.

$$P(\mathbf{s}, \mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{e^{\mathbf{w}^T \mathbf{F}(\mathbf{s},\mathbf{y},\mathbf{x}) + \mathbf{u}^T \mathbf{G}(\mathbf{s},\mathbf{y},\mathbf{x})}}{Z(\mathbf{x})} \quad (3)$$

$$\mathbf{F}(\mathbf{s}, \mathbf{y}, \mathbf{x}) = \sum_{k=1}^{K} \mathbf{f}(s_k, y_k, \mathbf{x}) \quad (4)$$

$$\mathbf{G}(\mathbf{s}, \mathbf{y}, \mathbf{x}) = \sum_{k=1}^{K} \mathbf{g}(y_k, y_{k-1}, \mathbf{x}) \quad (5)$$

3. CHORD RECOGNITION FEATURES

Given a segment s and chord y, we will use the following notation:

- $s.Notes$, $s.N$ = the set of notes in the segment s.
- $s.Events$, $s.E$ = the sequence of events in s.
- $e.len$, $n.len$ = the length of event e or note n, in quarters.
- $e.acc$, $n.acc$ = the accent value of event e or note n, as computed by the `beatStrength()` function in Music21 [1]. This is a value that is determined based on the metrical position of n, e.g. in a song written in a 4/4 time signature, the first beat position would have a value of 1.0, the third beat 0.5, and the second and fourth beats 0.25. Any other eighth note position within a beat would have a value of 0.125, any sixteenth note position strictly within the beat would have a value of 0.0625, and so on.
- $y.root$, $y.third$, and $y.fifth$ = the triad tones of the chord y.
- $y.added$ = the added note of chord y, if y is an added tone chord.

We use the following heuristics to determine whether a note n from a segment s is a figuration note with respect to a candidate chord label y:

1. **Passing**: There are two anchor notes n_1 and n_2 such that: n_1's offset coincides with n's onset; n_2's onset coincides with n's offset; n_1 is one scale step below n and n_2 is one step above n, or n_1 is one step above n and n_2 one step below; n is not longer than either n_1 or n_2; the accent value of n is strictly smaller than the accent value of n_1; at least one of the two anchor notes belongs to segment s; n is non-harmonic with respect to chord y, i.e. n is not equivalent to the root,

[1] http://web.mit.edu/music21

third, fifth, or added note of y; both n_1 and n_2 are harmonic with respect to the segments they belong to.

2. **Neighbor**: There are two anchor notes n_1 and n_2 such that: n_1's offset coincides with n's onset; n_2's onset coincides with n's offset; n_1 and n_2 are both either one step below or one step above n; n is not longer than either n_1 or n_2; the accent value of n is strictly smaller than the accent value of n_1; at least one of the two anchor notes belongs to segment s; n is non-harmonic with respect to chord y; both anchor notes are harmonic with respect to the segments they belong to.

3. **Suspension**: Note n belongs to the first event of segment s. There is an anchor note m in the previous event (last event in the previous segment) such that: m and n have the same pitch; n is either tied with m (i.e. held over) or m's offset coincides with n's onset (i.e. restruck); n is not longer than m; n is non-harmonic with respect to chord y, while m is harmonic with respect to the previous chord.

4. **Anticipation**: Note n belongs to the last event of segment s. There is an anchor note m in the next event (first event in the next segment) such that: n and m have the same pitch; m is either tied with n (i.e. held over) or n's offset coincides with m's onset (i.e. restruck); n is not longer than m; n is non-harmonic with respect to chord y, while m is harmonic relative to all other notes in its event.

Futhermore, because we are using the weak semi-CRF features shown in Equation 4, we need a heuristic to determine whether an anchor note is harmonic whenever the anchor note precedes the current segment. The heuristic simply looks at the other notes in the event containing the anchor note: if the event contains 2 or more other notes, at least 2 of them need to be consonant with the anchor, i.e. intervals of octaves, fifths, thirds, and their inversions; if the event contains just one other note, it has to be consonant with the anchor.

We emphasize that the rules mentioned above for detecting figuration notes are only approximations. We recognize that correctly identifying figuration notes can also depend on subtler stylistic and contextual cues, thus allowing for exceptions to each of these rules.

Equipped with this heuristic definition of figuration notes, we augment the notation as follows:

- $s.Fig(y)$ = the set of notes in s that are figuration with respect to chord y.
- $s.NonFig(y) = s.Notes - s.Fig(y)$ = the set of notes in s that are not figuration with respect to y.

Some of the segment-label features introduced in this section have real values. Given a real-valued feature $f(s, y)$ that takes values in $[0, 1]$, we discretize it into $K+2$ Boolean features by partitioning the $[0, 1]$ interval into a set of K subinterval bins $\mathcal{B} = \{(b_{k-1}, b_k] | 1 \leq k \leq K\}$. For each bin, the corresponding Boolean feature determines whether $f(s, y) \in (b_{k-1}, b_k]$. Additionally, two Boolean features are defined for the boundary cases $f(s, y) = 0$ and $f(s, y) = 1$. For each real-valued feature, unless specified otherwise, we use the bin set $\mathcal{B} = [0, 0.1, ..., 0.9, 1.0]$.

3.1 Segment Purity

The segment purity feature $f_1(s, y)$ computes the fraction of the notes in segment s that are harmonic, i.e. belong to chord y:

$$f_1(s, y) = \frac{\sum_{n \in s.Notes} \mathbf{1}[n \in y]}{|s.Notes|}$$

The duration-weighted version $f_2(s, y)$ of the purity feature weighs each note n by its length $n.len$:

$$f_2(s, y) = \frac{\sum_{n \in s.Notes} \mathbf{1}[n \in y] * n.len}{\sum_{n \in s.Notes} n.len}$$

The accent-weighted version $f_3(s, y)$ of the purity feature weighs each note n by its accent weight $n.acc$:

$$f_3(s, y) = \frac{\sum_{n \in s.Notes} \mathbf{1}[n \in y] * n.acc}{\sum_{n \in s.Notes} n.acc}$$

The 3 real-valued features are discretized using the default bin set $\mathcal{B}$.

3.1.1 Figuration-Controlled Segment Purity

For each segment purity feature, we create a figuration-controlled version that ignores notes that were heuristically detected as figuration, i.e. replace $s.Notes$ with $s.NonFig(y)$ in each feature formula.

3.2 Chord Coverage

The chord coverage features determine which of the chord notes belong to the segment. The first 3 features refer to the triad notes:

$$\begin{aligned} f_4(s, y) &= \mathbf{1}[y.root \in s.Notes] \\ f_5(s, y) &= \mathbf{1}[y.third \in s.Notes] \\ f_6(s, y) &= \mathbf{1}[y.fifth \in s.Notes] \end{aligned}$$

A separate feature determines if the segment contains all the notes in the chord:

$$f_7(s, y) = \prod_{n \in y} \mathbf{1}[n \in s.Notes]$$

A chord may have an added tone $y.added$, such as a 4th, a 6th, or a 7th. If a chord has an added tone, we define two features that determine whether the segment contains the added note:

$$\begin{aligned} f_8(s, y) &= \mathbf{1}[\exists y.added \wedge y.added \in s.Notes] \\ f_9(s, y) &= \mathbf{1}[\exists y.added \wedge y.added \notin s.Notes] \end{aligned}$$

Through the first feature, the system can learn to prefer the added tone version of the chord when the segment contains it, while the second feature enables the system to learn to prefer the triad-only version if no added tone is in the segment. To prevent the system from recognizing added chords too liberally, we add a feature that is triggered whenever the total length of the added note in the segment is greater than the total length of the root:

$$alen(s,y) = \sum_{n \in s.Notes} \mathbf{1}[n = y.added] * n.len$$

$$rlen(s,y) = \sum_{n \in s.Notes} \mathbf{1}[n = y.root] * n.len$$

$$f_{10}(s,y) = \mathbf{1}[\exists y.added] * \mathbf{1}[alen(s,y) > rlen(s,y)]$$

The duration-weighted versions of the chord coverage features weigh each chord tone by its total duration in the segment. For the root, the feature would be computed as shown below:

$$f_{11}(s,y) = \frac{\sum_{n \in s.Notes} \mathbf{1}[n = y.root] * n.len}{\sum_{n \in s.Notes} n.len}$$

Similar features f_{12} and f_{13} are computed for the third and the fifth. The corresponding accent-weighted features f_{14}, f_{15}, and f_{16} are computed in a similar way, by replacing the note duration $n.len$ in the duration-weighted formulas with the note accent value $n.acc$.

The duration-weighted feature for the added tone is computed similarly:

$$f_{17}(s,y) = \frac{\mathbf{1}[\exists y.added] * \sum_{n \in s.Notes} \mathbf{1}[n = y.added] * n.len}{\sum_{n \in s.Notes} n.len}$$

Furthermore, by replacing $n.len$ with $n.acc$, we also obtain the accent-weighted version f_{18}.

An alternative definition of duration-weighted features is based on the proportion of the segment time that is covered by a particular chord note. The corresponding duration-weighted feature for the chord root is shown below:

$$f_{19}(s,y) = \frac{\sum_{e \in s.Events} \mathbf{1}[y.root \in e] * e.len}{\sum_{e \in s.Events} e.len}$$

Similar duration-weighted features normalized by the segment length are defined for thirds, fifths, and added notes.

All duration-weighted and accent-weighted features are discretized using the default bin set $\mathcal{B}$.

3.2.1 Figuration-Controlled Chord Coverage

For each chord coverage feature, we create a figuration-controlled version that ignores notes that were heuristically detected as figuration, i.e. replace $s.Notes$ with $s.NonFig(y)$ in each feature formula.

3.3 Bass

The bass note provides the foundation for the harmony of a musical segment. For a correct segment, its bass note often matches the root of its chord label. If the bass note instead matches the chord's third, fifth, or added dissonance, this may indicate that the chord is inverted. Thus, comparing the bass note with the chord tones can provide useful features for determining whether a segment is compatible with a chord label.

There are multiple ways to define the bass note of a segment s. One possible definition is the lowest note of the first event in the segment, i.e. $s.e_1.bass$. Comparing it with the root, third, fifth, and added tones of a chord results in the following features:

$$\begin{aligned} f_{20}(s,y) &= \mathbf{1}[s.e_1.bass = y.root] \\ f_{21}(s,y) &= \mathbf{1}[s.e_1.bass = y.third] \\ f_{22}(s,y) &= \mathbf{1}[s.e_1.bass = y.fifth] \\ f_{23}(s,y) &= \mathbf{1}[\exists y.added \wedge s.e_1.bass = y.added] \end{aligned}$$

An alternative definition of the bass note of a segment is the lowest note in the entire segment, i.e. $\min_{e \in s.E} e.bass$. The corresponding features will be:

$$\begin{aligned} f_{24}(s,y) &= \mathbf{1}[y.root = \min_{e \in s.E} e.bass] \\ f_{25}(s,y) &= \mathbf{1}[y.third = \min_{e \in s.E} e.bass] \\ f_{26}(s,y) &= \mathbf{1}[y.fifth = \min_{e \in s.E} e.bass] \\ f_{27}(s,y) &= \mathbf{1}[\exists y.added \wedge y.added = \min_{e \in s.E} e.bass] \end{aligned}$$

The duration-weighted version of the bass features weigh each chord tone by the time it is used as the lowest note in each segment event, normalized by the duration of the bass notes in all the events. For the root, the feature is computed as shown below:

$$f_{28}(s,y) = \frac{\sum_{e \in s.Events} \mathbf{1}[e.bass = y.root] * e.len}{\sum_{e \in s.Events} e.len}$$

Similar features f_{29} and f_{30} are computed for the third and the fifth. The duration-weighted feature for the added tone is computed as follows:

$$f_{31}(s,y) = \frac{\mathbf{1}[\exists y.added] * \sum_{e \in s.E} \mathbf{1}[e.bass = y.root] * e.len}{\sum_{e \in s.E} e.len}$$

The corresponding accent-weighted features f_{31}, f_{32}, f_{33}, and f_{34} are computed in a similar way, by replacing the bass duration $e.bass.len$ in the duration-weighted formulas with the note accent value $e.bass.acc$.

All duration-weighted and accent-weighted features are discretized using the default bin set $\mathcal{B}$.

3.3.1 Figuration-Controlled Bass

For each bass feature, we create a figuration-controlled version that ignores event bass notes that were heuristically detected as figuration, i.e. replace $e \in s.Events$ with $e \in s.Events \wedge e.bass \notin s.Fig(y)$ in each feature formula.

3.4 Chord Bigrams

The arrangement of chords in chord progressions is an important component of *harmonic syntax* [1]. A first order semi-Markov CRF model can capture chord sequencing information only through the chord labels y and y' of the current and previous segment. To obtain features that generalize to unseen chord sequences, we follow Radicioni and Esposito [9] and create chord bigram features using only a) the *mode*: major (M), minor (m), or diminished (d); b) the *added* note: none ($\emptyset$), fourth (4), sixth (6), or seventh (7); and c) the interval in semitones between the roots of the two chords.

$$g_1(y, y') = \mathbf{1}[y.mode = \{M, m, d\} \wedge y.added = \{\emptyset, 4, 6, 7\} \wedge y'.mode = \{M, m, d\} \wedge y'.added = \{\emptyset, 4, 6, 7\} \wedge |y.root - y'.root| = \{0, 1, ..., 11\}]$$

Note that $g_1(y, y')$ is a feature template that can generate (3 modes $\times$ 4 added)$^2\times$ 12 intervals = 1,728 distinct features. To reduce the number of features, we use only the *(mode.added)–(mode.added)'–interval* combinations that appear in the manually annotated chord bigrams from the training data.

4. CHORD RECOGNITION DATASETS

For evaluation, we used two chord recognition datasets:

1. BCHD: this is the Bach Choral Harmony Dataset, a corpus of 60 four-part Bach chorales that contains 5,664 events and 3,090 segments in total [9].

2. TAVERN: this is a corpus of 27 complete sets of theme and variations for piano, composed by Mozart and Beethoven. It consists of 63,876 events and 12,802 segments overall [2].

The BCHD corpus has been annotated by a human expert with chord labels, using the set of labels described in Section 2. Of the 144 possible labels, 102 appear in the dataset and of these only 68 appear 5 times or more. Some of the chord labels used in the manual annotation are enharmonic, e.g. C-sharp major and D-flat major, or D-sharp major and E-flat major. Reliably producing one of two enharmonic chords cannot be expected from a system that is agnostic of the key context. Therefore, we normalize the chord labels and for each mode we define a set of 12 canonical roots, one for each scale degree. When two enharmonic chords are available for a given scale degree, we selected the one with the fewest sharps or flats in the corresponding key signature. Consequently, for the major mode we use the canonical root set {C, Db, D, Eb, F, Gb, G, Ab, A, Bb, B}, whereas for the minor and diminished modes we used the root set {C, C#, D, D#, F, F#, G, G#, A, Bb, B}. Thus, if a chord is manually labeled as C-sharp major, the label is automatically changed to the enharmonic D-flat major. The actual chord notes used in the music are left unchanged. Whether they are spelled with sharps or flats is immaterial, as long as they are enharmonic with the root, third, fifth, or added note of the labeled chord.

The TAVERN dataset [2] currently contains 17 works by Beethoven (181 variations) and 10 by Mozart (100 variations). The themes and variations are divided into a total of 1,060 phrases, 939 in major and 121 in minor. The pieces have two levels of segmentations: chords and phrases. The chords are annotated with Roman numerals, using the Humdrum representation for functional harmony [3]. When finished, each phrase will have annotations from two different experts, with a third expert adjudicating cases of disagreement between the two. After adjudication, a unique annotation of each phrase is created and joined with the note data into a combined file encoded in standard **kern format. However, many pieces do not currently have the second annotation or the adjudicated version. Consequently, we only used the first annotation for each of the 27 sets. Furthermore, since our chord recognition approach is key agnostic, we developed a script that automatically translated the Roman numeral notation into the key-independent canonical set of labels used in BCHD. Because the TAVERN annotation does not mark added fourth or sixth notes, the only added chords that were generated by the translation script were those containing sevenths. This results in a set of 72 possible labels, of which 69 appear in the dataset.

Few other annotated chord recognition datasets exist. One of these is the Kostka-Payne corpus [4], a dataset of 46 brief excerpts compiled by David Temperley from Kostka and Payne's music theory textbook [4]. Several chord recognition systems have used this dataset in the past [8,9]. However, it is smaller than both BCHD and TAVERN, with 3,964 events and only 779 segments. Another dataset is Chris Harte's Beatles collection [3], containing annotations for 12 complete albums. Though this dataset is much larger in size, the chord labels are mapped to audio. We considered re-mapping these labels to MIDI files, but had difficulty finding accurate MIDI files for most Beatles songs.

5. EXPERIMENTAL EVALUATION

We implemented the semi-Markov CRF chord recognition system using a multi-threaded package [5] that has been previously used for noun-phrase chunking of informal text [7]. Following the experimental setting from [9], we evaluated the semi-CRF model on BCHD using 10-fold cross validation: the 60 Bach Chorales were split into 10 folds, and each fold was used as test data, with the other nine folds being used for training. We used the same parti-

[2] https://github.com/jcdevaney/TAVERN
[3] http://www.humdrum.org/Humdrum/representations/harm.rep.html
[4] http://www.cs.northwestern.edu/ pardo/kpcorpus.zip
[5] http://statnlp.org/research/ie/

tion into folds as that employed by Radicioni and Esposito [9], to enable comparison with their perceptron-trained HMM system, henceforth referred to as HMPerceptron. We also used their set of labels, consisting of the 102 chords observed in the dataset, which corresponds to 90 canonical chords. For each feature we computed its frequency of occurrence in the training data, using only the true segment boundaries and their labels. To speedup training and reduce overfitting, we only used features whose counts were at least 5. The performance measures were computed by pooling together the results from the 10 test folds. Table 1 shows the event-level and segment-level performance of the semi-CRF model, together with two versions of the HMPerceptron: HMPerceptron$_1$, for which we do enharmonic normalization both on training and test data, similar to the normalization done for semi-CRF; and HMPerceptron$_2$, which is the original system from [9] that does enharmonic normalization only on test data. When computing the segment-level performance, a predicted segment is considered correct only if both its boundaries and its label match those of the true segment.

System	Acc$_E$	P$_S$	R$_S$	F$_S$
semi-CRF	**83.16**%	**77.60**%	**73.48**%	**75.48**%
HMP-tron$_1$	80.30%	74.18%	69.76%	71.90%
HMP-tron$_2$	80.16%	70.24%	65.78%	67.94%

Table 1. Comparative results on the BCHD dataset, using Event-level accuracy (Acc$_E$) and Segment-level precision (P$_S$), recall (R$_S$), and F-measure (F$_S$).

The semi-CRF model achieves a 3% improvement in accuracy over the original HMPerceptron model, which corresponds to a 15% relative error reduction. The improvement in accuracy over HMPerceptron$_1$ is statistically significant at a p-value of 0.01, using a one-tailed Welch's t-test over the sample of 60 chorale results. The improvement in segment-level performance is even more substantial, with a 7.5% absolute improvement in F-measure over the original HMPerceptron model, and a 3.6% improvement in F-measure over the HMPerceptron$_1$ version, which is statistically significant at a p-value of 0.04, using a one-tailed Welch's t-test.

Error analysis revealed wrong predictions being made on chords that contained dissonances that spanned the duration of the entire segment (e.g. a second above the root of the annotated chord), likely due to an insufficient number of such examples during training. Manual inspection also revealed a non-trivial number of cases in which the authors disagreed with the manually annotated chords, e.g. some chord labels were clear mistakes, as they did not contain any of the notes in the chord. This further illustrates the necessity of building music analysis datasets that are annotated by multiple experts, with adjudication steps akin to the ones followed by TAVERN.

To evaluate on the TAVERN corpus, we created a test dataset from 6 Beethoven sets (*B063*, *B064*, *B065*, *B066*, *B068*, *B069*) and 4 Mozart sets (*K025*, *K179*, *K265*, *K353*). The remaining 11 Beethoven sets and 6 Mozart sets were used for training. All sets were normalized enharmonically before being used for training or testing. Table 2 shows the event-level and segment-level performance of the semi-CRF and HMPerceptron model on the TAVERN dataset. Despite the smaller number of chord labels (69 in TAVERN vs. 90 in BCHD), the results in Tables 1 and 2 show that chord recognition is substantially more difficult in the TAVERN dataset. The comparatively lower performance on TAVERN is likely due to the substantially larger number of figurations and higher rhythmic diversity of the variations compared to the easier, mostly note-for-note texture of the chorales. Error analysis on TAVERN revealed many segments where the first event did not contain the root of the chord. For such segments, HMPerceptron incorrectly assigned chord labels whose root matched the bass of this first event. Since a single wrongly labeled event invalidates the entire segment, this can explain the larger discrepancy between the event-level accuracy and the segment-level performance. In contrast, semi-CRF assigned the correct labels in these cases, likely due to its ability to exploit context through segment-level features, such as the chord root coverage feature f_4 and its duration-weighted version f_{11}.

System	Acc$_E$	P$_S$	R$_S$	F$_S$
semi-CRF	**77.47**%	**66.86**%	**60.35**%	**63.44**%
HMP-tron	60.55%	27.83%	23.21%	25.31%

Table 2. Comparative results on the TAVERN dataset, using Event-level accuracy (Acc$_E$) and Segment-level precision (P$_S$), recall (R$_S$), and F-measure (F$_S$).

6. CONCLUSION AND FUTURE WORK

We presented a semi-Markov CRF approach to chord recognition that does joint segmentation and labeling of tonal music in symbolic form. Compared to event-level tagging approaches based on HMMs or linear CRFs, the segmental CRF approach has the advantage that it can accommodate features that consider all the notes in a candidate segment. This capability was shown to be especially useful for music with complex textures that diverge from the simpler note-for-note structures of the Bach chorales. On the more difficult TAVERN corpus, the semi-CRF substantially outperformed a previous system based on event-level tagging, thus validating empirically the suitability of joint segmentation and labeling for chord recognition.

Manually engineering good features for chord recognition is a cognitively demanding and time consuming process that requires music theoretical knowledge and that is unlikely to lead to optimal sets of features, especially when complex features are involved. In future work we plan to investigate automatic feature extraction using recurrent neural networks (RNN) that preserve the semi-Markov property, such as the recently proposed segmental RNNs [5].

7. ACKNOWLEDGMENTS

We would like to thank Patrick Gray for his help with preprocessing the TAVERN corpus. We also greatly appreciate the prompt help provided by Daniele P. Radicioni and Roberto Esposito. Their detailed responses to our questions and willingness to share the original BCHD dataset and the HMPerceptron code enabled us to conduct a more thorough experimental comparison with their system. Finally, we would like to thank the anonymous reviewers for their insightful comments on the initial draft of this paper.

8. REFERENCES

[1] E. Aldwell, C. Schachter, and A. Cadwallader. *Harmony and Voice Leading*. Schirmer, 4th edition, 2011.

[2] Johanna Devaney, Claire Arthur, Nathaniel Condit-Schultz, and Kirsten Nisula. Theme and variation encodings with roman numerals (TAVERN): A new data set for symbolic music analysis. In *International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[3] Christopher Harte. *Towards Automatic Extraction of Harmony Information from Music Signals*. PhD thesis, Queen Mary University of London, August 2010.

[4] Stefan Kostka and Dorothy Payne. *Tonal Harmony*. McGraw-Hill, 1984.

[5] Liang Lu, Lingpeng Kong, Chris Dyer, Noah A. Smith, and Steve Renals. Segmental recurrent neural networks for end-to-end speech recognition. In *INTERSPEECH*, San Francisco, CA, September 2016.

[6] H. John Maxwell. An expert system for harmonizing analysis of tonal music. In Mira Balaban, Kermal Ebcioğlu, and Otto Laske, editors, *Understanding Music with AI*, pages 334–353. MIT Press, Cambridge, MA, USA, 1992.

[7] Aldrian Obaja Muis and Wei Lu. Weak semi-Markov CRFs for noun phrase chunking in informal text. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 714–719, San Diego, California, June 2016. Association for Computational Linguistics.

[8] Bryan Pardo and William P. Birmingham. Algorithms for chordal analysis. *Computer Music Journal*, 26(2):27–49, July 2002.

[9] Daniele P. Radicioni and Roberto Esposito. BREVE: an HMPerceptron-based chord recognition system. In *Advances in Music Information Retrieval*, pages 143–164. Springer Berlin Heidelberg, 2010.

[10] Christopher Raphael and Josh Stoddard. Harmonic analysis with probabilistic graphical models. In *International Society for Music Information Retrieval Conference (ISMIR)*, 2003.

[11] Thomas Rocher, Matthias Robine, Pierre Hanna, and Robert Strandh. Dynamic chord analysis for symbolic music. In *International Computer Music Conference, ICMC*, 2009.

[12] Sunita Sarawagi and William W. Cohen. Semi-Markov Conditional Random Fields for information extraction. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, pages 1185–1192, Cambridge, MA, USA, 2004. MIT Press.

[13] Ricardo Scholz and Geber Ramalho. Cochonut: Recognizing complex chords from midi guitar sequences. In *International Conference on Music Information Retrieval (ISMIR)*, pages 27–32, Philadelphia, USA, September 14-18 2008.

[14] David Temperley and Daniel Sleator. Modeling meter and harmony: A preference-rule approach. *Computer Music Journal*, 23(1):10–27, January 1999.

[15] Terry Winograd. Linguistics and the computer analysis of tonal harmony. *Journal of Music Theory*, 12(1):2–49, 1968.