AutoRhythmGuitar: Computer-aided Composition for Rhythm Guitar in the Tab Space

Matt McVicar
National Institute of Advanced Industrial Science and Technology (AIST). Tsukuba, Japan
mattjamesmcvicar@gmail.com

Satoru Fukayama
National Institute of Advanced Industrial Science and Technology (AIST). Tsukuba, Japan
s.fukayama@aist.go.jp

Masataka Goto
National Institute of Advanced Industrial Science and Technology (AIST). Tsukuba, Japan
m.goto@aist.go.jp

ABSTRACT

We present AutoRhythmGuitar, a simple computer-aided composition model which algorithmically composes realistic rhythm guitar tablature. AutoRhythmGuitar takes as input a downbeat-synchronised chord sequence and generates a digital score in both traditional notation and tablature. Our model is data-driven, trained from digital tablatures obtained from the internet. By varying this training data, we are able to model individual rhythm guitarists' rhythmic and melodic styles. Algorithmic evaluation of our system reveals that it effectively models style, whilst a qualitative analysis by the authors confirms that the resulting tablatures are realistic and, for the most part, playable.

1. INTRODUCTION

In this paper, we consider the problem of computer-aided composition for the guitar. In popular music, guitar parts can broadly be split into *rhythm* parts (mostly outlining the main harmony and rhythmic pulse) and *lead* parts (mostly melody lines and solo breaks) – we focus on composition of rhythm guitar parts in the current paper. The main data flow and processes of our system are outlined in Figure 1.

1.1 Motivation

Our motivation for investigating this problem is two-fold. First, we wish to investigate if an analysis of guitarist performance reveals significant musician-specific trends in rhythmic and melodic devices. Furthermore, we believe automatic generation of guitar parts in a particular style could be used as a pedagogical aid, to help amateur musicians learn different approaches to playing over a given chord sequence. It is worth noting at this point that the generation of complex rhythm guitar parts, in the style of a given player, is a non-trivial task (see Sub. 2.2) and also currently beyond the capabilities of software such as 'Band in a Box' [1].

[1] http://www.pgmusic.com/bbwin.htm

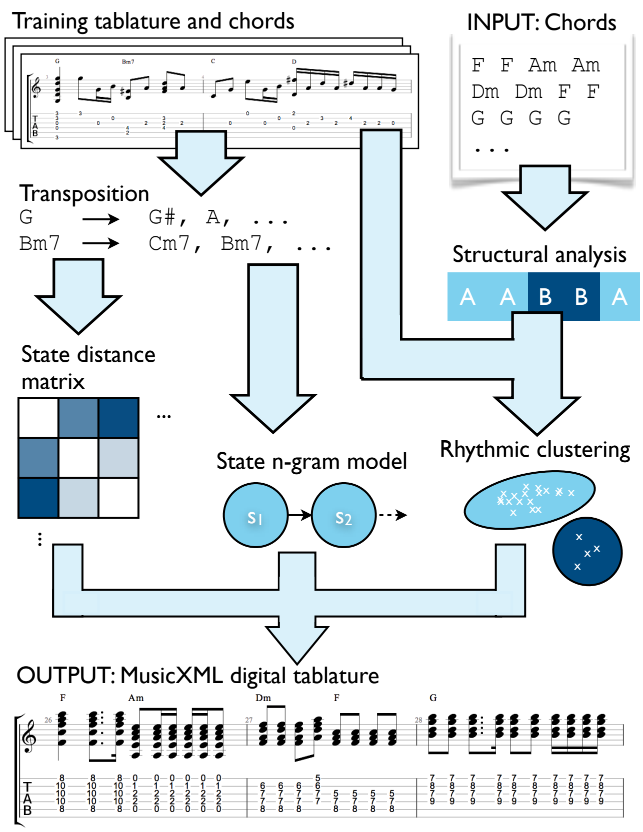

Figure 1. AutoRhythmGuitar's main processes. The system takes as input a downbeat-synchronised chord sequence and conducts a structural analysis. The detected segments are then combined with rhythms from a set of training tablatures and clustered into an appropriate number of groups. Meanwhile, $n-$gram models and state distances are calculated from the training data. The output of these three processes are then used to construct a digital tablature in MusicXML format.

1.2 Challenges and proposed solutions

There are many obstacles to overcome when devising an algorithmic composition method for the guitar. The first challenge is that unconstrained algorithmic composition is extremely challenging given the variety and complexity of music, and even with human aid (Computer-Aided Composition, CAC) developing methods which generalise well to unseen situations remains problematic. We tackle these challenges in the current work by having users of AutoRhythmGuitar input a chord sequence to the model (see Figure 1), and by using resources from the web, transposing the data to maximize the generalisation potential (see Subsections 4.4 and 5.1).

Second, we are not currently aware of any CAC systems which are guitar-specific. If existing general–purpose CAC models are used for our task, the resulting piece may not be playable on the guitar, owing to sudden jumps around the neck. This would therefore necessitate an arrangement of the piece (see 2.2 for a definition of this term). Furthermore, these systems do not incorporate appealing features of the instrument such as hammer-ons, pull-offs, or slides. In this paper, we identify these as guitar-specific challenges and solve them by composing rhythm guitar music directly in the tablature space (see 2.1). Furthermore, we use algorithmic means to ensure that the resulting music is playable (4.4), with models for the previously-mentioned ornaments built into the model (4.5).

Finally, we observed (see 4.1) that professional rhythm guitarists use a high degree of repetition within songs, and that this repetition appears to be grouped into structures. Without prior knowledge of musical structure, existing techniques would fail to replicate this behaviour. An analysis of the input chord sequence is therefore conducted in this work to make our output tablatures *structurally consistent* (see 4.5).

1.3 Paper structure

In Section 2, we discuss tablature notation and outline what we believe to constitute a rhythm guitarist's style. Section 3 then provides an overview of the relevant literature in computer-aided composition and automatic guitar arrangement. Our compositional model is presented in Section 4, and evaluated and analysed in Section 5. Finally, conclusions and suggestions for further work are outlined in Section 6.

2. BACKGROUND

2.1 Guitar tablature

The pitch ranges for guitar strings significantly overlap, so that for most pitches there exists no unique playing position (string and fret number) for a given note. This one-to-many relationship means that guitarists must make a decision on where best to fret each note on the instrument to minimise overall fretting hand movement, which can be challenging for beginner guitarists [1]. For this reason tablature notation (or simply 'tab', plural 'tabs'), which explicitly specifies the strings and frets on which notes are to be played, was developed. Examples of tabs alongside traditional musical notation are shown in Figure 2. Due to its unambiguous nature, tab is extremely popular amongst musicians, and it is for these reasons and with our pedagogical motivation in mind that the current study focuses on producing tablature output for rhythm guitar.

2.2 Styles of rhythm guitar playing

Despite the discussion above, it should be noted that the many-to-one mapping of fingering positions to musical score offers practitioners of the guitar great freedom in hand positioning and note selection given an underlying chord, and as such can be considered a creative benefit of the instrument. We postulate that professional guitarists develop a preference for certain chord shapes and fingerboard positions, and that this can be considered an aspect of their style (see examples below).

To avoid confusion with existing terminology [2], we introduce the term *melodic voicing* to mean the free choice of notes and fingerboard positions a guitarist makes when composing a rhythm guitar part for a given chord. Illustrative examples showing the melodic voicings five popular guitarists have taken to playing over a C major chord are shown in Figure 2.

In the first measure, Eric Clapton plays a C 'fifth' chord (no third) in third position followed by a melodic break in the A minor pentatonic scale. The second measure shows Jimi Hendrix adding a ninth to the chord in eighth position with a leading melody to the D chord which follows (not shown). Jimmy Page takes a straightforward 'hard rock' approach in third position, whilst the last two guitarists (Keith Richards, The Rolling Stones; Slash, Guns N' Roses) opt for open position melodic voicings, but show two distinct approaches; the former strumming three or four note chords with alternating bass, the latter arpeggiating the chord in a typical rock ballad style.

It is precisely these aspects of rhythm guitar playing which will be attempting to model and imitate in this work. We next discuss the literature relevant to the current study.

3. EXISTING WORK

3.1 Computer-aided composition

Algorithmic composition can be described as the process of using a sequence of rules to combine musical parts into a composition [2] and has a rich and varied research history (see, for example, [3, 4] or the survey [5]), of which an interesting subset is Computer-Aided Composition (CAC) [6, 7, 8]. In this scenario, the compositional task is split between the computer and a human expert.

In line with the increase in availability of digital musical information, data-driven approaches to CAC have gained popularity in recent years. Widmer [9], and Schwanauer and Levitt [10] were both early adopters of the data-driven approach in the harmonization of a given melody. Conklin et al. [11] examined the prediction and generation of chorale music from examples. Dubnov et al. [12] investigated the modelling of musical style, learning from MIDI input in a wide variety of styles. Pachet and various collaborators [13, 14] have investigated the use of Markov chains for generation of novel content, with constraints to avoid plagiarism.

[2] *fingering decision*: mapping a score to tab, *arrangement*: minimally modifying a piece initially not written for guitar to make it playable [1].

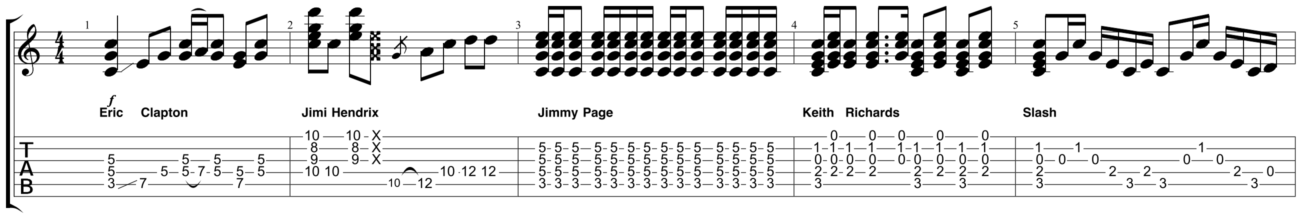

Figure 2. Examples of rhythm guitar styles taken by five professional guitarists when faced with a C major chord measure. Musical score is shown above, tablature below. Samples are taken (left to right) from "Badge" (Clapton/Harrison), "Fire" (Hendrix), "Immigrant Song" (Page/Plant), "Wild Horses" (Jagger/Richards), "Knockin' on Heaven's Door" (Dylan, arranged by Guns N' Roses). Notation used: / = slide (glissando), ‿ / ⁀ = hammer-on/pull-off, X = muted string

3.2 Automatic guitar fingering and arrangement

Sayegh first considered the problem of automatic arrangement for stringed instruments in 1989 [15], introducing an *optimum path paradigm* solution to the fingering problem, which was later extended by Radicioni et al. [16] to minimise phrase-level, rather than global, fingering difficulty. The latter model was evaluated on a single classical guitar piece of twenty-five measures, consisting of single notes (no chord tones), and was judged to be similar to the arrangement provided by a musical expert.

The *path difference learning* algorithm was introduced by Radisavljevic and Driessen [17], which learns the weight costs of a particular playing style based on labelled tabs. On a set of seven classical guitar pieces, the number of fingering errors when compared to a human arrangement dropped from 101 to 11 on the training set as the model converged, but they noted that results did not generalise well to unseen data due to a lack of training examples.

Genetic algorithms have been explored by Tuohy et al. [18, 19] as a means of efficiently exploring the large search space created in the fingering decision problem, in which the majority of the generated tablature coincided with human-made annotations on selections from 34 guitar pieces of varying style. Recently, Yazawa et al. [20] also investigated the transcription of synthesized MIDI audio into playable guitar tablature by the use of playability constraints.

Finally, an Input-Output Hidden Markov Model has been suggested by Hori et al. [1] to assign fingerings to a given piece, where the hidden states represented physical positions of the fretting hand, and the observed states represented the notes produced. Model output was compared to commercial software on three pieces totalling seven measures, although no quantitative evaluation was performed.

4. MODEL DESCRIPTION

4.1 Coupling of rhythm and melody

To gain insight into how best to approach rhythm guitar composition, we begun by investigating some examples produced by professionals. We obtained digital guitar tabs for a selection of guitarists from GuitarProTab.net [3] . These tabs were exported to MusicXML via the GuitarPro software [4] to facilitate computational analysis. The rhythm for each measure was encoded as a length 16 vector $\mathbf{r}$ representing the note type at each sixteenth note. Measures which contained note durations shorter than this or tuplets were omitted from analysis.

We classified each sixteenth note as either an onset; held (sustained) note; rest; or muted note, denoting these rhythmic states as $[0, 1, 2, 3]$ respectively, so that $\mathbf{r} \in \{0, 1, 2, 3\}^{16}$. We then defined rhythmic similarity between pairs $\mathbf{r}_1, \mathbf{r}_2$ using the normalised Hamming similarity [21]:

$$S_{\text{rhythm}}(\mathbf{r}_1, \mathbf{r}_2) = \frac{1}{16} \sum_{i=1}^{16} \mathbb{1}(\mathbf{r}_1^i = \mathbf{r}_2^i). \qquad (1)$$

For melodic similarity, we collected the fretboard positions of every note or chord into a list of (string, fret) pairs, calling this a model *state*. For example, the state corresponding to the first quarter note in measure 1 in Figure 2 would be $[(3, 5), (4, 5), (5, 3)]$. Given that the number of states in a measure may differ and we are interested in the overlap of states and not their order in particular, we opted for the Jaccard index to define melodic similarity between two measures $\mathbf{M}_1, \mathbf{M}_2$:

$$S_{\text{melody}}(\mathbf{M}_1, \mathbf{M}_2) = \frac{|\mathbf{M}_1 \cap \mathbf{M}_2|}{|\mathbf{M}_1 \cup \mathbf{M}_2|},$$

where $|\cdot|$ indicates set cardinality and the intersection/union for measures $\mathbf{M}_1, \mathbf{M}_2$ is taken over states in the measures. We then plotted the rhythmic and melodic similarities in a Self Similarity Matrix (SSM), a selection of which can be seen in Figure 3. It can be see from Figure 3 that rhythm guitar compositions typically feature a large amount of repetition, and that similarities in rhythm (below main diagonal) and pitch (above main diagonal) are strongly correlated. This coupling is easily understood from the perspective of musical structure: it seems that rhythm guitarists employ distinct rhythmic and melodic patterns in sections such as *verse*, *refrain*, or *chorus*.

To this end, the first stage of our processing is to perform a structural analysis of the input chord sequence, which we assume contains cues on the structural landscape of the target song. This information will then be used to assign rhythms and melodic voicings (see Figure 1).

[3] `http://www.gprotab.net/index.php`

[4] `http://www.guitar-pro.com/en/index.php`

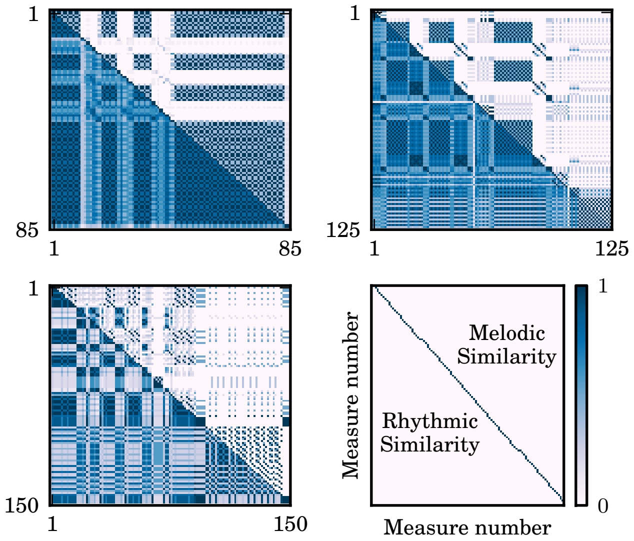

Figure 3. Self Similarity Matrices (SSM) for rhythmic (below main diagonal) and melodic (above main diagonal) similarity for a selection of three tabs. Top row: "Layla" (Clapton/Gordon), "Jumping Jack Flash" (Jagger/Richards). Bottom row: "Paradise City" (Rose, Slash, Stradlin/McKagen, Adler). Dark areas indicate high similarity.

4.2 Chord segmentation algorithm

We employ a novelty-based approach to detecting structure in our input symbolic chord sequence, adapting the approach by Foote [22]. Our algorithm takes as input a text file of M lines – one for each measure in the song. Each line describes the chords in a measure, which we assume to be in common time (4,4) and at the sixteenth-note resolution. The first stage of pre-processing is to label any measures which contain only *no chord* (silence etc.) as a unique segment type.

An $M \times M$ self-similarity matrix $\mathbf{S}$ is then computed, with similarity between the two length 16 vectors defined by Hamming similarity (Equation 1). We then pass an $n \times n$ binary checkerboard matrix $\mathbf{C}$ through the diagonal of $\mathbf{S}$, with the novelty at time t calculated as

$$\text{Novelty}(t) = \sum_{i=t-n/2}^{t+n/2} \sum_{j=t-n/2}^{t+n/2} \mathbf{C}_{i,j} \times \mathbf{S}_{i,j}.$$

The resulting novelty curve is then normalised to $[0, 1]$, and values which exceed the σ^{th} percentile selected as segment boundaries.

In informal testing, we found that this technique had high recall but poor precision, since the novelty in a close neighbourhood of true segment boundaries often exceeded the σ^{th} percentile. To counteract this behaviour, we discarded any segment boundaries at t which had another boundary with higher novelty within $[t-n/2, t+n/2]$. Each segment between boundaries was then labelled as a new segment.

Finally, we considered pairs of segments which were an integer multiple length of each other for merging (assigning the same segment label). We merged segments

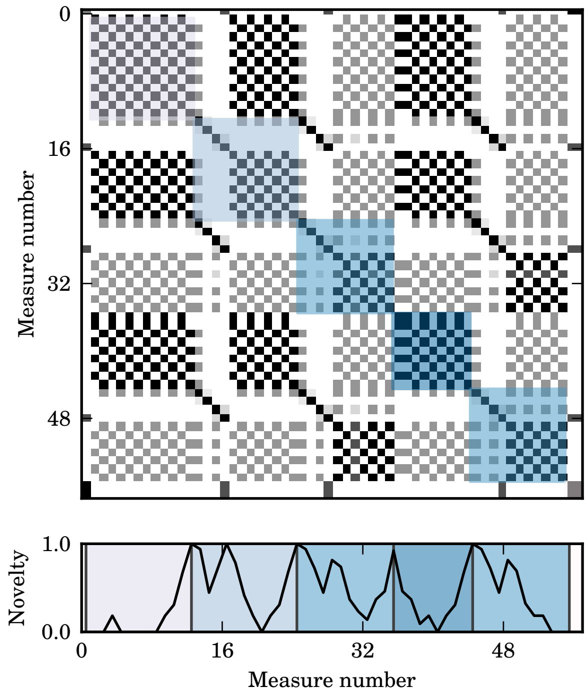

Figure 4. Example of our chord segmentation algorithm on "Imagine" (Lennon). Top: self-similarity matrix, with dark shades indicating high similarity. Below: the resulting novelty curve when passing an 8×8 checkerboard through the diagonal of this matrix. The segments as detected by the algorithm in Subsection 4.1 are overlaid, with segments of the same label having the same shade.

$\mathbf{s}_1 = [t_1, \ldots, t_1 + l]$ and $\mathbf{s}_2 = [t_2, \ldots, t_2 + k \times l]$ if each of the k subsequences

$$\{[t_2, \ldots, t_2 + l], \ldots, [t_2 + (k-1) \times l, \ldots, t_2 + k \times l\}$$

has Hamming similarity with $\mathbf{s}_1$ greater than τ. An example of our algorithm for the chords to "Imagine" (Lennon) is shown in Figure 4, where in this example and throughout the remainder of this paper we set the parameters $n = 8$, $\sigma = 75$, $\tau = 0.75$. Our algorithm has labelled the first and last two measures as 'No chord' segments, and identified five main segments, two of which (three and five) have been assigned the same label. These segments constitute contiguous chorus and verses, which were unfortunately not merged with the second main segment due to a segment length difference of one measure (12 vs. 11). Improving and evaluating this simple segmentation algorithm is part of our planned future work.

After segments in the target chord sequence have been automatically analysed, the segments and labels are fed into AutoRhythmGuitar's two main processes: rhythm assignment and melodic voicing assignment. These are detailed in the following two Subsections.

4.3 Rhythm assignment

As per the examples in Subsection 4.1, we assume time is discretized to sixteen-note resolution in common time, and denote the rhythm of a measure as $\mathbf{r} \in \{0, 1, 2, 3\}^{16}$ (recall the rhythmic states: note onset, held note, rest, muted note).

The total number of unique rhythmic measures under this model is 4^{16}, although we believe the number of rhythms of this type used by popular music guitarists to be far fewer than this in practice. For this reason, in this paper we take an *example-based* approach to rhythm assignment. That is to say, the generated rhythms will come directly from our training data. However, the question remains as how to assign one of the training rhythms to each of the test measures.

In tackling this problem, we assume that guitarists have a number of rhythmic styles at their disposal, with each style consisting of a set of similar rhythms. For example, one rhythmic style might consist mostly of rests with the occasional muted sixteenth note, whilst another might consist only of quarter and half note onsets. To discover these groupings, we therefore clustered our training rhythms.

To set the number of desired clusters c, we turn to our input chord sequence, which we assume has been segmented into s distinct segment types via the algorithm in 4.2. It is clear to us that in order to maximise the rhythmic distinction between segments (thus emulating the behaviour seen in 4.1), we should set $c = s$.

To see this, suppose $c < s$. Then there are fewer rhythmic clusters than distinct segments, and some segments would have the same rhythmic style, which we consider undesirable. Conversely, if $c > s$ then there are more rhythmic clusters than segments and some rhythmic styles would have to be discarded. Furthermore, the rhythmic clusters in this scenario will be less well separated than if $c \leq s$.

The rhythms obtained from the training data were therefore clustered into s clusters. We opted for the spectral clustering algorithm, which takes an input an arbitrary distance measure between data points (for which we used the Hamming distance, $1-$ Equation (1)). Seeing no other obvious way to proceed, we matched the resulting rhythm cluster j to chord segment i randomly. However, in sampling from rhythm cluster j, we sample an example rhythm $\mathbf{r}$ from cluster j with probability proportional to the frequency of $\mathbf{r}$ in j. This ensures that more common rhythms within a cluster are more likely to appear in the output.

4.4 Melodic voicing assignment

Through the processes in Subsections 4.2 and 4.3, we have segmented the target chord sequence into labelled segments and have assigned rhythms to each measure. Our task now is to assign a *state* (recall: a model state is a list of string and fret pairs) to each note onset.

4.4.1 n–gram modelling

Recall one of our goals is to create *playable* guitar tablature (see challenges, Subsection 1.2). To this end, whilst a chord is constant within a measure we use n–gram modelling, a technique popular for modelling many time-dependent stochastic processes including automatic speech recognition [23] and chord estimation [24].

For each chord in the training data, we therefore collected *initial* and *bigram* counts for each state. A melodic voicing assignment for a chord y is then produced by first sampling from the initial distribution for y, followed by a biased random walk on the state distribution for y. However, before normalising our counts to form probability distributions, we first transposed our data, as detailed below.

4.4.2 Transposition

In order to maximally exploit the available training data and our model's generalisation potential, all training chords and states (except those which contained open strings) were transposed up and down the guitar neck to increase the number of state-to-state transitions witnessed. The underlying assumption which facilitates this is that a guitarist's melodic voicing approach is pitch-independent. In other words, that each of the first three guitarists in Figure 2 would equally likely play the same melodic patterns a fret higher if presented with a C♯ major chord instead of C major chord, analogously for one fret down / B major chord etc.

We therefore transposed each state in the initial and bigram counts for every chord down the neck until the lowest fret played was equal to 0, and up the neck until the highest fret played was equal to a pre-defined maximum, which we set to be 12. The counts for the original state were then added to the counts for the transposed chords. After this was completed for every chord and state, the resulting counts were normalised to sum to unity.

This 'transposition trick' means that data for chords in more common guitar keys (the 'open' keys: G, C, D for example) may be used to train models for keys in which there is likely to be less data (A♭, B♭ etc. which do not feature convenient open string pitches in standard tuning), all the while meaning it is likely that each bigram with non-zero probability is playable (since it appeared at least once, perhaps transposed, in the training data). Crucially, it also allows AutoRhythmGuitar to generalise to chord labels not seen in the training data (addressing one of the challenges of this work, see 1.2), so long as the unseen chord *type* (major, diminished etc) appears at least once.

4.4.3 State-to-state distance

Using the above techniques we found that our model produced playable tab whilst a chord was constant, but that between chords unplayable sequences were sometimes introduced, due to the model sampling from the initial distribution for the next chord with no knowledge of the current hand position. To counteract this behaviour we introduced a state-to-state distance inspired by Hori et al. [1].

The distance proposed in [1] takes into account the fingering arrangements of both states as well as the time allowed for the change, allowing for greater movement if time allows. They define the distance D to s_2 from s_1 given t (elapsed time) via a modified Laplace distribution:

$$D_{\text{state}}(s_2|s_1, t) = \frac{1}{2t} \exp\left(-\frac{|I_2 - I_1|}{t}\right) \frac{1}{1 + I_2} \frac{1}{1 + W_2} \frac{1}{1 + N_2}, \tag{2}$$

where t indicates the time since the last note was fretted, I_1 and I_2 are the index finger positions of states s_1 and s_2, W_2

is the 'fret span' of s_2 (max. fret minus min. fret), and N_2 is the number of fingers used in s_2.

We use Equation (2) as above with the following simplifications, given we had no fingering data for our states. We set I_1, I_2 to be the minimum fret for each state. We assume the number of fingers used is equal to the number of non-open string notes in the state. This assumption is valid for all single notes and most chords, except those which use barres.

Finally then, we set the probability p of the first state of a chord y being s given a time lapse of t equal to a weighted sum of the initial probability of s given y, $P_{\text{ini}}(s|y)$ and the distance from to this state from the previous model state:

$$p = \alpha P_{\text{ini}}(s|y) + (1-\alpha) D_{\text{state}}(s|\text{previous state}, t).$$

In our experiments for this paper we set $\alpha = 0.5$ without any attempt to optimise performance.

4.5 Guitar-specific ornaments

After the rhythms and states for our target chord sequence have been assigned, we added guitar-specific ornaments to enhance the realism of AutoRhythmGuitar's output, addressing some of the challenges mentioned in Subsection 1.2. Specifically, we allow a hammer-on (note sounded by 'hammering' from one fret on a string to a higher fret without plucking/picking), pull-off (analogously) or slide (glissando) between states, with the probabilities of these special transitions between states occurring learnt from the data using the method detailed in Subsection 4.4 (including transposition). Note that these ornaments may be learnt in an artist-specific manner, using the exact same methodology as for the state transitions, by selectively sampling our training data.

4.6 Structural consistency

Finally, if the current measure is part of a segment for which content has already been generated, AutoRhythmGuitar simply repeats this content. This is conducted to emulate the behaviour seen in Figure 3, and to produce a *structurally consistent* composition.

5. EXPERIMENTS

5.1 Training Data

We choose five well-known guitarists (Jimi Hendrix, Keith Richards, Jimmy Page, Slash, Eric Clapton) to train our model, and downloaded ten digital tabs (GuitarPro format files) for each guitarist (song titles available on our Vimeo page, see Subsection 5.3). The guitarists and tabs were chosen according to popularity (measured by number of available tabs) and quality (similarity to audio recording and author knowledge) with songs chosen which were (at least predominantly) in common time and standard tuning (or down one semitone, which is easily transposed). Where more than one tab was available for a song, the most accurate or complete tab was chosen. Each digital tab was then converted to MusicXML format via GuitarPro for analysis.

Chord annotations and hierarchical beat structure (downbeat and main pulse) for each song were then obtained automatically using the online service Songle [25] using the official YouTube video as input, and were subsequently checked and edited for correctness by an expert musician.

5.2 Algorithmic evaluation

In this Subsection, we investigate if our model is able to model rhythmic and melodic rhythm guitar styles. This is realised by training models for our five guitarists of choice and comparing summaries of the distributions obtained. If the distributions are significantly non-homogeneous, it gives evidence that each model represents a different style (if indeed each guitarist has a unique style).

To this end, we trained five models and computed summary distributions as follows. Each rhythm $\mathbf{r} \in \{0,1,2,3\}^{16}$ in the training set was converted to a categorical '1-of-4' vector $\hat{\mathbf{r}} \in \{0,1\}^{64}$. These vectors per measure were then summed over the songs and normalised per sixteenth note, resulting in a vector for each artist which represents the probability of a note onset, held note, rest, or muted note at each of the sixteen metric positions. For each chord, we computed the probability of each state associated with this chord occurring by simply counting and normalising.

Distributions $P(x), Q(x)$ were then compared based on the Kullback-Leibler (KL) divergence:

$$D_{\text{KL}}(P||Q) = \sum_i \ln\left(\frac{P(i)}{Q(i)}\right) P(i). \tag{3}$$

For rhythmic similarity, we used the symmetric KL-divergence:

$$D_{\text{KL}}(P,Q) = D_{\text{KL}}(P||Q) + D_{\text{KL}}(Q||P). \tag{4}$$

For melodic similarity however, we conditioned Equation (3) on the probability of each chord occurring:

$$D_{\text{KL}}(P(s|y)||Q(s|y)) = \sum_y P(y) \sum_s \ln\left(\frac{P(s|y)}{Q(s|y)}\right) P(s|y),$$

where $P(y)$ is the probability of chord y occurring and s are the states for chord y. This divergence was then made symmetric analogously to Equation (4). The results of these experiments can be seen in Figure 5. We see few areas of self-similarity and a fairly high degree of homogeneity, indicating that the distributions are 'far apart', giving evidence that rhythm guitarists have a distinct style, which AutoRhythmGuitar has effectively modelled. In both Subfigures the higher distances in row/column 4 suggest that Keith Richards' rhythmic and melodic style are the most unique seen in the dataset (see also 5.3).

5.3 Qualitative analysis

Since our system outputs MusicXML, it can be easily imported into a variety of existing software packages for synthesis. To assess the quality and playability of the tabs our system generates, we therefore trained one model for each of the five guitarists listed above and imported our model's output into GuitarPro. We chose "Imagine" (Lennon) as a

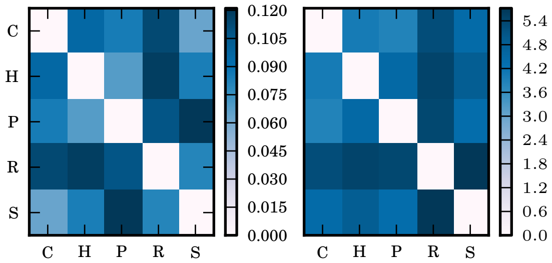

Figure 5. Rhythmic/Melodic (left/right) KL-divergence between guitarists. ['C', 'H', 'P', 'R', 'S'] = Eric Clapton, Jimi Hendrix, Jimmy Page, Keith Richards, Slash. The difference in magnitude between the plots is due to the melodic model having many more states than the rhythmic model.

test case as it is a well-known song with an interesting array of chords which does not feature a guitar part.

We synthesized the output of our model in GuitarPro with an appropriate backing track consisting of piano, drums, melody line and cello, and selected an appropriate guitar tone for each artist. The results are available for viewing at our Vimeo page [5], which we encourage the reader to visit whilst reading the remainder of this Subsection. A small number of examples can also be seen in Figure 6. Our comments on the output (which can also be found in the video descriptions) make up the remainder of the current Subsection.

5.3.1 Jimi Hendrix

Jimi Hendrix's unique rhythm guitar style appears to be modelled effectively using AutoRhythmGuitar. Throughout the first verse we see partial chords (over the C chord, see Figure 6) and melodic phrases using an added ninth (F chord). An unexpected benefit of implementing muted notes also occurs in this verse: the muted note (measure 3) allows the player time to move back to first position. In the chorus, we see an A minor shape (measure 14) not exploited by many guitarists, although AutoRhythmGuitar has used it to minimise the amount of fretting hand movement required. The remainder of the chorus features typical partial chords and some interesting passing tones typical of his style.

Subsequent verses feature melodic phrases with many guitar-specific ornaments such as slides (see Figure 6) and hammer-ons. The final sections (from measure 26) feature extensive use of rapid muted notes (measure 31), his 'thumb over the top' technique (measure 27) and more partial chords (measure 30).

5.3.2 Keith Richards

Suspended chords are commonly used by Keith Richards, and this is reflected from the outset in this model output (Csus4 over C chord, measure 2, see also Figure 6). The slightly unusual jump from twelfth fret to first position (measure 3) is a result of the α parameter too strongly enforcing the fretboard locality constraint, when moving from measure 2 to measure 3. The chorus and second verse are both harmonically sound and also playable, and feature a major chord voicing not used by any other guitarists in our dataset (measure 20).

The advantage of using a state-to-state distance is clearly highlighted in measures 27-28, however: with hardly any fretting hand movement, the player is able to provide a melodic voicing for three distinct chords. The chord voicing for the F chord in measure 26 with the additional fifth note on the top E string is also unique to Keith Richards in our dataset, and is repeated over the E7 measures in this song. The final unique section (measures 37–45) feature a more minimal rhythm guitar approach, with just single notes or diads highlighting the underlying chords.

5.3.3 Jimmy Page

AutoRhythmGuitar's output in the style of Jimmy Page begins with melodic passages over the C chord and a challenging fretboard movement over the F chord, meaning that some manual tuning of the parameter α might be required to increase playability for this piece. However, in the subsequent verse these issues are not seen, and the chorus shows the first case of Page's arpeggio style (measure 26).

Measure 30 then introduces the non-diatonic B♭ note, although the result is in fact harmonious. The concluding verse again uses arpeggios, this time over an entire measure (measure 37 and Figure 6). Note again AutoRhythmGuitar's ability to select an appropriate F chord voicing (eighth position) to closely match the previous measure's final state.

5.3.4 Slash

Slash's approach to rhythm guitar playing is typical of the hard rock style, and this is evident immediately from this output. The rhythmic approach is exclusively eighth notes in the first verse, with melodic voices consisting of either a repeated root note or fifth chord (see measure 2, Figure 6). Note again that there are many ways in which these melodic voices could be played, but that AutoRhythmGuitar has selected a pair which involve minimal fretting hand movement. The first chorus then introduces some muted notes in between this same basic approach (measure 16).

The second verse continues this theme, but unfortunately contains an almost impossible jump from open position C to two G notes an octave apart (measure 18). The refrain (beginning measure 30) features some slight dissonance (C and B notes over a C major chord, measure 31) but this could be an aspect of Slash's playing, since it appears in our training data. The final unique section is very minimal, featuring single sustained root notes.

5.3.5 Eric Clapton

The first verse of this output features simple fifth chords and diads (see Figure 6), with some additional percussive muted notes. In the chorus, the A minor chord is arpeggiated, and there is a pleasant melodic line in harmonised sixths, although some dissonance is introduced with the E♭ over the F chord. This we discovered was due to us incorrectly labelling a dominant 7 chord as a major chord in the training data.

[5] `https://vimeo.com/user25754596/videos`

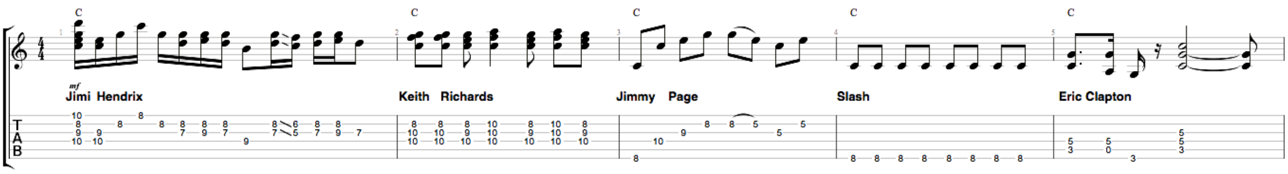

Figure 6. Example AutoRhythmGuitar output, showing the different rhythm guitar styles our system is able to emulate.

The first measure of the second verse is challenging to play due to the jump to the tenth fret. In measure 30, we see a weakness of our system: it has filled the entire measure with an F fifth chord, neglecting the underlying G chord which follows. This is due to the dissociation of rhythm and pitch in our model, which assigns rhythms for a measure without knowledge of the position of any chord changes contained within. The remainder of the song is both harmonically consonant and playable.

6. CONCLUSIONS AND FUTURE WORK

We presented AutoRhythmGuitar, a system which produces realistic, structurally consistent and (for the most part) playable guitar tablature in the style of a given artist from a chord sequence input. Our contributions in this work were as follows. First, we used an input chord sequence to constrain the algorithmic composition problem, with models per chord trained using data from the web. Second, we created realistic and playable music by composing directly in the tab space, using $n-$gram models, a state-to-state distance and guitar-specific ornaments to increase the playability and realism of our output. Our final contribution was the segmentation of the input chord sequence, in order to optimally decide the number of training rhythms and produce a structurally consistent composition.

In future work, we would like to explore ways of overcoming some of the limitations of our system, including: increasing the temporal resolution, improving and evaluating our segmentation algorithm, methods for optimising the distance weight α, as well as methods for generating lead guitar parts. We are also interested in developing our algorithms for use by amateur musicians in the general public, possibly as part of a web service.

Acknowledgments

This work was supported in part by CREST, JST.

7. REFERENCES

[1] G. Hori, H. Kameoka, and S. Sagayama, "Input-output HMM applied to automatic arrangement for guitars," *Journal of Information Processing*, vol. 21, no. 2, pp. 264–271, 2013.

[2] G. Papadopoulos and G. Wiggins, "AI methods for algorithmic composition: a survey, a critical view and future prospects," in *AISB Symposium on Musical Creativity*, 1999, pp. 110–117.

[3] L. Hiller and L. Isaacson, *Experimental Music; Composition with an electronic computer*. Greenwood Publishing Group Inc., 1979.

[4] D. Cope, "A computer model of music composition," in *Machine models of music*. MIT Press, 1992, pp. 403–425.

[5] C. Ames, "The markov process as a compositional model: a survey and tutorial," *Leonardo*, pp. 175–187, 1989.

[6] G. Nierhaus, *Algorithmic Composition*. Springer-Verlag/Wien, 2009, ch. 11.

[7] F. Pachet, "The continuator: Musical interaction with style," *Journal of New Music Research*, vol. 32, no. 3, pp. 333–341, 2003.

[8] S. Fukayama, D. Saito, and S. Sagayama, "Assistance for novice users on creating songs from japanese lyrics," in *Proc. ICMC*, 2012, pp. 441–446.

[9] G. Widmer, "Qualitative perception modeling and intelligent musical learning," *Computer Music Journal*, vol. 16, no. 2, pp. 51–68, 1992.

[10] S. Schwanauer and D. Levitt, *Machine models of music*. MIT Press, 1993.

[11] D. Conklin and I. Witten, "Multiple viewpoint systems for music prediction," *Journal of New Music Research*, vol. 24, no. 1, pp. 51–73, 1995.

[12] S. Dubnov, G. Assayag, O. Lartillot, and G. Bejerano, "Using machine-learning methods for musical style modeling," *Computer*, vol. 36, no. 10, pp. 73–80, 2003.

[13] F. Pachet and P. Roy, "Markov constraints: steerable generation of Markov sequences," *Constraints*, vol. 16, no. 2, pp. 148–172, 2011.

[14] A. Papadopoulos, P. Roy, and F. Pachet, "Avoiding plagarism in Markov sequence generation," in *AAAI*, 2014.

[15] S. Sayegh, "Fingering for string instruments with the optimum path paradigm," *Computer Music Journal*, vol. 13, no. 3, pp. 76–84, 1989.

[16] D. Radicioni, L. Anselma, and V. Lombardo, "A segmentation-based prototype to compute string instruments fingering," in *Proc. Conference on Interdisciplinary Musicology, Graz*, 2004.

[17] A. Radisavljevic and P. Driessen, "Path difference learning for guitar fingering problem," in *Proc. ICMC*, vol. 28, 2004.

[18] D. Tuohy and W. Potter, "A genetic algorithm for the automatic generation of playable guitar tablature," in *Proc. ICMC*, 2005, pp. 499–502.

[19] D. Tuohy, "Creating tablature and arranging music for guitar with genetic algorithms and artificial neural networks," Master's thesis, University of Georgia, 2006.

[20] K. Yazawa, D. Sakaue, K. Nagira, K. Itoyama, and H. Okuno, "Audio-based guitar tablature transcription using multipitch analysis and playability constraints," in *Proc. ISASSP*, 2013, pp. 196–200.

[21] G. Toussaint, "A comparison of rhythmic similarity measures." in *Proc. ISMIR*, 2004, pp. 242–245.

[22] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *Proc. Multimedia and Expo*, vol. 1. IEEE, 2000, pp. 452–455.

[23] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[24] K. Yoshii and M. Goto, "A vocabulary-free infinity-gram model for nonparametric bayesian chord progression analysis." in *Proc. ISMIR*, 2011, pp. 645–650.

[25] M. Goto, K. Yoshii, H. Fujihara, M. Mauch, and T. Nakano, "Songle: A web service for active music listening improved by user contributions." in *Proc. ISMIR*, 2011, pp. 311–316.