A Greedy Approach to Music Voice Separation Using Hidden Markov Models

Edson E. da Silveira*, Shigeki Sagayama**

* Meiji University, Tokyo, Japan
edson.contact@gmail.com
** Meiji University, Tokyo, Japan
sagayama@meiji.ac.jp

Abstract. This research, in the area of music information retrieval, describes a probabilistic method of separating notes into different voices for music pieces encoded as MIDI files. The proposed solution represents the music as a sequence of chords that are analyzed individually in a linear fashion. The problem of voice assignment is modelled as an HMM problem at each chord. Given a list of all previously given assignments, the model suggests the most likely voice assignments for the current chord. The model was tested on a dataset of 111 J. S. Bach pieces with *F-measure* scores for J. S. Bach's 15 Inventions (99.17) and J. S. Bach's The Well-Tempered Clavier 48 Preludes (95.28) and 48 Fugues (96.97) being reported. The results show improvement over currently available methods on the same dataset.

Keywords: Dynamic Programming, HMM, Voice Separation, Music Information Retrieval

1 Introduction

Music commonly makes use of the concept of simultaneous stream of notes that the listener cognitively interprets as separate voices. This resulting polyphonic structure is therefore critical when notating and comprehending music. In the realm of Music Information Retrieval, separating notes into voices is very often a necessary step for other relevant tasks. As interest in this field of study increases, computational solutions have grown in importance in the last decade. This can often be a daunting endeavor as other MIR tasks require voices to be monophonic. Out of convenience we use the definition of voice as a monophonic sequence of non-overlapping in this work. However, other definitions exist as discussed in [1] and MIDI ground truth data are not always available in this fashion. In addition, without human assistance, voice-crossings, long rests, and the estimation of the number of voices are, among other problems, a great challenge to voice separation methods. In this paper we present a probabilistic model that separates notes into voices for music scores encoded as MIDI files. As in some works, the score is seen as a sequence of chords. However, unlike others where HMM is applied for observations ordered in time, usually a sequence of chords, our model uses HMM over

the simultaneous notes in the chords where the ordering is instead given by pitch value. This convenient use of HMM enables us to break down the problem as a sequence of small HMM problems interconnected by a global list of voice assignments that provides the knowledge of all previously given assignments.

2 Related Studies

Voice separation has been an area of constant improvement in recent years with much of the current studies taking as basis music perception-derived voice leading principles from Huron's study in the field [2]. Important definitions from that work such as pitch proximity and temporal continuity are often used, alone or in combination, as the core musical concepts for the great majority of the voice separation methods. The former, pitch proximity, indicates that successive notes that are close in pitch maintain the coherence of an auditory stream, or voice as the term is used in this study. The latter, temporal continuity, indicates the role of temporal proximity in successive notes in evoking the perception of continuity in an auditory stream. Approaches which apply the above musical concepts on the other hand, have been more diverse.

Chew and Wu [3] describes a *contig* approach where a piece of music is divided into segments with a constant number of voices observed. This method also estimates the total number of voices in a piece as the number of notes of its largest chord. It does not allow voice-crossings as they are uncommon. This creates a limiting factor when analyzing pieces where they do occur.

Karydis, Nanopoulos, Papadopoulos, and Cambouropoulos [4] also propose a method (updated in [5] and [6]) assuming polyphonic in contrast to monophonic voices as most methods.

Duane and Pardo [8], like [3] also divides the music into segments. Those are then analyzed through constraint optimization before being joined to the others using sequence alignment techniques. Again, as in [3], voice crossings are not permitted and are the source of most errors as observed by the authors. In their approach the notes present in each segment can be represented as vertices and edges connect pair of notes according to a weight function forming voices.

Jordanous [7] uses a machine learning approach to avoid pitfalls such as undetected voice crossings in more rule-based approaches as in [3]. This training-heavy approach is intended to be more flexible to music outside the classical realm where such perceptual principles may not hold as much relevance. It also searches for periods of clear harmonic structure as starting points. The method used does not take temporal proximity in the notes of a voice into consideration. Instead, the probabilities are mainly derived from the proximity in terms of pitch. Results are given only for fugues of predefined number of voices, 3 and 4-voice fugues from J. S. Bach's The Well-Tempered Clavier.

Valk, Weyke and Benetos [9] also use a machine learning technique for a very practical problem. They propose the use of a neural network model for voice separation to achieve their goal of separating voices in old lute tablature. Their aim is to transcribe those pieces into modern music notation for easier accessibility of the corpus. Because of the focus on the lute as an instrument they assume the total number of voices in a piece to be at most five.

McLeod and Steedman [10] apply Hidden Markov models with a modified Viterbi algorithm that can be applied for real time data. They use the idea of average voice pitch, which we also use, to better represent the changes in the average pitch of a voiceover time. One drawback in their approach comes from defining all possible groups of assignments as hidden states, creating an ever exponentially increasing state space. This requires modification of the standard Viterbi algorithm in order to reduce the search-space.
Gray and Bunescu [11], as with [9], present another neural network solution with their greedy neural network model. As in [10] and this paper, they do not set the number of voices a priori to enable the application in a diverse dataset but still achieve results very similar results to the ones found in [9], that also uses neural networks.

3 Proposed Method

3.1 Preliminaries

Our method uses the MIDI file representation of score as an input, extracting the following information of each MIDI note: *onset*, *offset*, *pitch number* and *track*. The last, *track*, is only used when evaluating the method as it provides the ground truth voice assignments. Each MIDI *pitch number* represents a difference of half-tone musically with the number 60 being equivalent to the middle C (C4).
The underlying music perception concepts used to calculate the probabilities are pitch proximity and temporal continuity, explained briefly in the previous section, that are applied with a certain degree of liberty. We also come from the assumption that global optimization is not always advantageous and can often hinder real time applications (also a consideration in the McLeod and Steedman model [10]).

3.2 The Model

For the proposed model, a piece of music in seen as a sequence of note sets, or chords, $M = \{C_1, C_2, \dots, C_T\}$ where each chord C_i is defined as a sequence of all notes $n_1 \dots n_m$ of equal onset time and ordered by decreasing pitch value. That is, for any two notes $n_i, n_{i+1} \in C_i$ we have:

$$On(n_i) = On(n_{i+1}) \tag{1}$$

$$Pitch(n_i) \leq Pitch(n_{i+1}) \tag{2}$$

With $On(n)$ denoting the onset time of a note n and $Pitch(n)$ the MIDI number of the corresponding note.

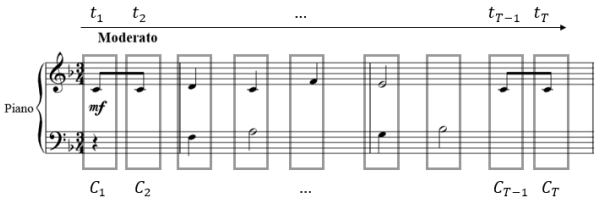

Fig. 1. Music score as a sequence of chords.

The chords are analyzed consecutively, and every note $n_i \in C_t$ given a voice assignment. At each step the method receives as input a list $V_{t-1} = \{v_{1_{t-1}}, v_{2_{t-1}}, \dots, v_{n_{t-1}}\}$, initially empty, of all previous voice assignments where $v_{i_{t-1}}$ represents a monophonic voice as of time $t-1$. This chord analysis outputs an updated list $V_t = \{v_{1_t}, v_{2_t}, \dots, v_{n_t}\}$ that includes the voice assignments for the notes in C_t. This updated list is then given as input for the analysis of C_{t+1} and the process is repeated until all chords are analyzed, and all notes assigned to a voice.

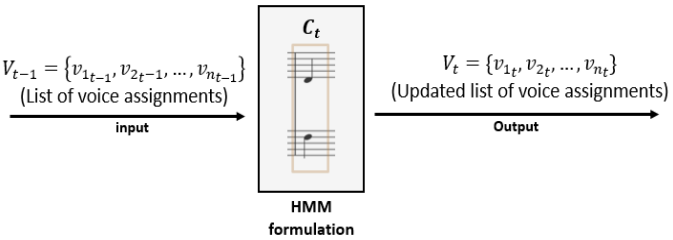

Fig. 2. Analysis of a chord.

Also, we make use of the average voice pitch function proposed by McLeod and Steedman. They define the average pitch of a voice as the *"weighted average of its l most recent notes, where each successive note is weighed more heavily than the previous note by a factor of two"* [10]. This a convenient approach as voices can change slightly over time. New voices are also introduced in consideration to the position of the other voices at the time they start. Although no great improvement in our evaluation metrics were observed by taking this approach over the use of pitch of the first note of the voice instead, in some cases this definition can clear ambiguities as to which voice is the closest to an observed note. As we believe this can be relevant in other datasets we like to make use of it. We describe the average pitch function in (3).

$$AP(v) = \frac{\sum_{i=0}^{\min(l,|v|)} \left(2^i * Pitch\left(n_{|v|-i}\right)\right)}{\sum_{i=1}^{\min(l,|v|)} 2^i} \tag{3}$$

Chord Analysis

Every chord C_t has its notes assigned to voices through the use of a Hidden Markov model. In our formulation, the observed sequence is the set of m notes $n_i, \dots, n_m \in C_t$.

The hidden states are a group of $n + m$ states containing the n voices $v_1, \dots, v_n \in V_t$ in addition to m hidden states $v_{n+1}, \dots, v_{n+m}$ representing the potential new voices for each of the m observed notes.

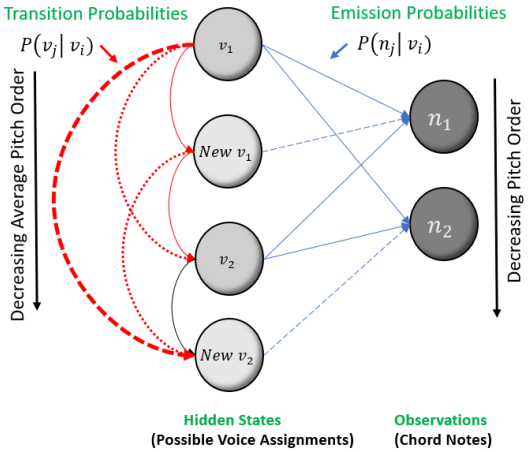

Fig. 3. HMM example with voice assignments as hidden states and chord notes as observations. There are two existing voices and two possible new ones (one for each note observed).

Initial State Probabilities

The distinction above between existing and potential new voices is important when defining the initial state probabilities (4). The latter is given half the probability of the former, where we work on the assumption that pre-existing voices should be given preference over possible new ones. Although the probabilities may not add up to 1, those values can be normalized and do not pose issues. The same is true for the transitions and emissions probabilities explained further below.

$$P(v_i) = \begin{cases} \dfrac{1}{n+m}, & if\ i \leq n \\ \dfrac{1}{2(n+m)}, & if\ i > n \end{cases} \tag{4}$$

Transition Probabilities

The state transition probabilities (5), in accordance with the initial probabilities, penalize any transitions to states representing possible new voices. As the observed notes are ordered by decreasing pitch value, transitions only to states of lower average pitch preserves monophony and discourages voice-crossings. This way, the lack of certain transitions observed in Fig.3 signifies that the targeted voice has a higher average pitch and the transition is therefore disallowed.

$$P(v_j|v_i) = \begin{cases} \frac{1}{(n+m)}, if\ j \leq n, AP(v_j) \leq AP(v_i) \\ \frac{1}{2(n+m)}, if\ j > n, AP(v_j) \leq AP(v_i) \\ 0, \quad otherwise \end{cases} \quad (5)$$

Emission Probabilities

Likewise, emission probabilities differ for states representing potential new voices. Those have their probabilities set to 0 to all but its corresponding observed note to maintain logical consistency. In this last case, the probability is given by (6), where b is an adjusted parameter. The more voices are added the lower is the probability given by (6).

$$P(n_j|\ v_i) = 1 - \frac{|V|}{|V| + b} \quad (6)$$

The emission probabilities for existing voices on the other hand, are calculated as the weighted sum of each music principle score used in this study (7) where $Pp(n_j, v_i)$, $Tc(n_j, v_i)$ and $Nr(n_j, v_i)$ correspond respectively to the *pitch proximity*, *temporal continuity* and *note recurrence* functions. $Off(n)$ in (8) represents the offset of a given note n and $Last(v_i)$ the last note n assigned to voice v_i. w_1, w_2, w_3 are the weights that are currently given equal values. Variable z given by (8) maintains temporal consistency by not allowing notes to be assigned to voices that would result in overlaps in terms of time.

$$P(n_j|\ v_i) = \left(w_1 * Pp(n_j, v_i) + w_2 * Tc(n_j, v_i) + w_3 * Nr(n_j, v_i)\right) * z \quad (7)$$

$$z = \begin{cases} 0, if\ Off(Last(v_i)) > On(n_j) \\ 1, if\ Off(Last(v_i)) \leq On(n_j) \end{cases} \quad (8)$$

Each music principle score function in (7) returns a value between in the interval [0,1] with a user adjusted parameter, a_1, a_2, a_3 that regulates the function's drop off.
The first, pitch proximity (9), gives a score according to the pitch distance between the observed note and the average pitch of the voice v_i (10).

$$Pp(n_j|\ v_i) = 1 - \frac{x}{x + a_1} \quad (9)$$

$$x = \left|AP(v_i) - Pitch(n_j)\right| \quad (10)$$

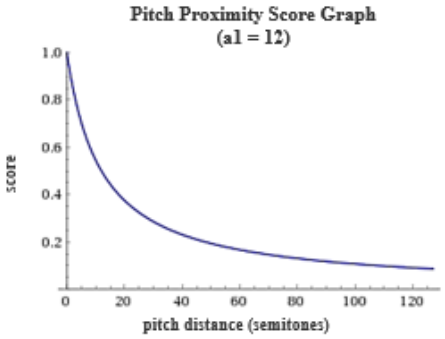

Fig. 4. Pitch proximity scores according to pitch distance. Parameter a_1 is initialized with the value used for our results.

The second, the temporal continuity (11), is the score given to the distance, in MIDI ticks, between the offset of voice v_i's last note and the onset of the observed note (12).

$$Tc(n_j | v_i) = 1 - \frac{x}{x + a_2} \tag{11}$$

$$x = \frac{|Off(Last(v_i)) - On(n_j)|}{100} \tag{12}$$

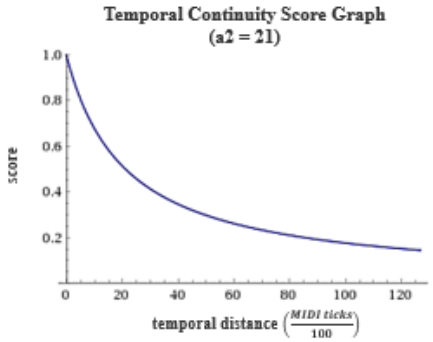

Fig. 5. Temporal continuity scores according to temporal distance. Parameter a_2 is initialized with the value used for our results.

Third and last, the note recurrence score (13) further emphasizes pitch proximity by giving scores according to the number of notes in v_i of the equal pitch value as the observed note n_i. This number is represented by function (14).

$$Nc(n_j | v_i) = \frac{x}{x + a_3} \tag{13}$$

$$x = Count(v_i, n_j) \tag{14}$$

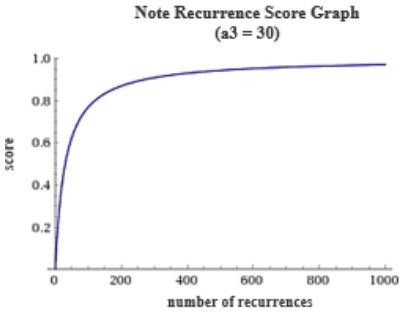

Fig. 6. Note recurrence scores according to number of recurrences. Parameter a_3 is initialized with the value used for our results.

Updating List of Assignments

The most likely assignments for each HMM is given by the standard Viterbi algorithm [12]. This result updates the global list of voice assignments V_{t-1} given as input. For any note assigned to a voice that didn't previously exist, a new voice is added to V_{t-1} and the note is added to it. The updated list is then given as input for the analysis of the next chord where the process is repeated. As such, the final solution V_T containing all voice assignments is equivalent to the set containing the solutions given by the Viterbi algorithm for each chord C_t. That is, if the most likely assignments given by the Viterbi for a chord C_t can be represented as

$$A^* = (v_1^*, \dots, v_n^*, \dots, v_{n+m}^*) = \underset{(v_1,\dots,v_{n+m})}{\operatorname{argmax}} P(v_1, \dots, v_{n+m} \mid n_1, \dots, n_m), \tag{15}$$

then the sequence $A_{t=1}^*, A_{t=2}^*, \dots, A_{t=T}^*$ is our final solution containing all voice assignments given by the method.

4 Evaluation

4.1 Dataset

For evaluation we use a dataset of MIDI files obtained from *www.musedata.com* containing the ground truth voice assignments. However, the data available is not completely monophonic and in some cases whole chords are grouped in the same track. This most commonly occurs at the last chord of a piece. As it diverges from our definition of voice as a monophonic sequence of non-overlapping notes, some misassignments are expected when using the data. We tested the oft-used set of 15 J. S. Bach 2-part inventions (BWV 772-786), as well as the 48 preludes and fugues from The Well-Tempered Clavier Book I & II (BWV 846-893). Although not found as a whole in other

studies, we thought as relevant the inclusion of all the 48 preludes from The Well-Tempered Clavier Book I & II for future reference.

4.2 Results

Except for the preludes, as no comparable data was found, we report our results using the information retrieval metrics of *precision*, *recall* and *F-measure*. As our *precision* and *recall* definitions match that of *soundness* and *closeness* in [9], something also noted and used in [7], we consider them as such when calculating and reporting their *F-measure* results for comparison purposes. The 10 fugues, also used in [9] and [11] consist of the first five 3 and 4-voice fugues from The Well-Tempered Clavier Book I, being therefore a subset of our dataset.
Function weights w_1, w_2 and w_3 from (7), parameter b from (6), music principle functions' parameters a_1, a_2 and a_3 from (9) (11) (13) respectively, average pitch function parameter l, were set by trial and error with the used values shown by table 1.

TABLE 1: MODEL PARAMETERS

Parameter	a_1	a_2	a_3	b	w_1	w_2	w_3	l
Value	12	21	30	0.5	1.5	1.3	0.4	2

Table 2 shows the results obtained by the method in our metrics as well as the standard deviation of the *F-measure* score. Each piece is calculated individually. Finally, in tables 3, 4 and 5 we compare the results with other studies that use similar datasets.

TABLE 2: MODEL RESULTS

Corpus	Precision	Recall	F-measure	St. Dev.
15 Inventions	99.28	99.06	99.17	0.64
WTC I (24 Fugues)	97.18	97.01	97.09	1.73
WTC II (24 Fugues)	96.89	96.81	96.85	1.56
WTC I & II (48 Fugues)	97.03	96.91	96.97	1.65
WTC I (24 Preludes)	96.31	93.53	94.76	6.29
WTC II (24 Preludes)	95.73	95.94	95.81	4.55
WTC I & II (48 Preludes)	96.01	94.73	95.28	5.51
10 Fugues	97.30	96.84	97.07	1.43

TABLE 3: THE WELL-TEMPERED CLAVIER

Corpus	Model	F-measure
WTC I (24 fugues)	Our model	97.09
	McLeod and Steedman [10]	97
	Duane and Pardo [8]	93
WTC II (24 fugues)	Our model	96.85
	McLeod and Steedman [10]	96
	Duane and Pardo [8]	92

TABLE 4: 10 FUGUES

Corpus	Model	F-measure
10 Fugues	Our model	97.07
	Gray and Bunescu [11]	93.87
	de Valk, Weyde and Benetos [9]	93.74

TABLE 5: 15 INVENTIONS

Corpus	Model	F-measure
15 Inventions	Our model	99.17
	McLeod and Steedman [10]	99
	Duane and Pardo [8]	95

It's important to note that the results reported are hampered by a few outliers that bring the overall accuracy down. This is especially true for the preludes as noted by the standard deviation, with pieces such as BWV 853 and 866 preludes scoring 72.83 and 80.08 respectively in the *F-measure*. The main reason for those low figures lies in the ground truth. As some simultaneous notes are assigned to the same voice in some chords, the model will diverge from the ground truth assignments. Although we mentioned previously that this most commonly occurs at the last chord, in pieces where the use of large chords is more predominant this is a recurring problem. In all these cases the model creates voices in excess causing a string of errors in the assignments to follow. In Fig. 7, an excerpt from BWV 866 prelude from The Well-Tempered Clavier Book I, we observe an instance of several notes belonging to the same voice according to the ground truth data. The piece is, otherwise, very non-polyphonic in texture with the voices alternating throughout the score.

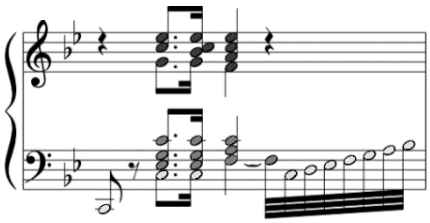

Fig. 7. Bar 13 from BWV 866 prelude with different shades of gray representing how the ground truth data used separates the voices.

5 Final Considerations

The model introduced in this paper proposes an alternative way of applying HMMs for hybrid, non-dynamic programming solutions. Unlike other machine learning solutions, where the chords are seen as the HMM system's observations, our model shows the use of HMMs as a convenient mathematical tool for calculating individual chord's note assignments with the notes themselves as the observations. We also make use of what we called 'note recurrence' that contributed to the pitch proximity principle that is usually used.

The results observed corroborate our approach with competitive results against currently available models, noting slightly higher scores than those found by McLeod and Steedman [10], the highest scoring model found with our dataset. However, their use of HMM as noted by the authors, requires modifications to the Viterbi algorithm due to a large and potentially infinite state-space. On the other hand, they provide support to real time input which were not considered in this paper and their model seems more robust in dealing with early stream error propagation. In our model, error propagation in early stages derive mostly from a non-optimal starting point, which is a common problem in several other methods. As with most studies, the model's definition of voice is another limitation when dealing with polyphonic voices. This aggravates error propagation issues that can lead to the creation of non-existing voices as discussed in the results section. Likewise, voice crossings can still cause some difficulties as a better way of identifying them is necessary. In addition, the flexibility added by not fixing the number of notes a priori can also lead to under or over-estimation of the number of voices. The model rarely adds more than one extra voice and even then, it's usually the result of whole chords being assigned to the same voice in the ground truth data.
Going forward, we believe more discussion regarding non-monophonic voices may prove invaluable for further improvement of the currently available methods. In the future, we plan further refinement of the music principles functions presented as well as the introduction of new ones from a pattern recognition perspective. In addition, we would like to introduce a less simplistic way of choosing the starting point for the method, possibly by identifying sections where the harmony is clearer as it has proved effective in other studies [3] and the use of a broader dataset, both in terms of classical composers as well as popular music.

References

[1] D. Rafailidis, A. Nanopoulos, E. Cambouropoulos, and Y. Manolopoulos, "Detection of Stream Segments in Symbolic Musical Data," *Proceedings of the 9th International Society for Music Information Retrieval Conference*, pp 83–88, Philadelphia, PA, 2008.

[2] D. Huron, "Tone and Voice: A Derivation of the Rules of Voice-Leading from Perceptual Principles," *Music Perception*, 19(1):1–64, 2001.

[3] E. Chew and X. Wu., "Separating Voices in Polyphonic Music: A Contig Mapping Approach," *Computer Music Modeling and Retrieval: 2nd International Symposium*, pp 1–20, 2004.

[4] I. Karydis, A. Nanopoulos, A. N. Papadopoulos, and E. Cambouropoulos, "VISA: The Voice Integration/Segregation Algorithm," *Proceedings of the 8th International Society for Music Information Retrieval Conference*, pp 445–448, Vienna, Austria, 2007.

[5] D. Rafailidis, E. Cambouropoulos, and Y. Manolopoulos," *Musical Voice Integration/Segregation: VISA Re- visited. In Proceedings of the 6th Sound and Music Computing Conference*, pp 42–47, Porto, Portugal, 2009.

[6] Makris, Dimos; Ioannis Karydis; Emilios Cambouropoulos, “VISA^3: Refining the Voice Integration/Segregation Algorithm,” *Proceedings of the Sound and Music Computing Conference 2016, SMC 2016,* Hamburg, Germany, 2016.

[7] A. Jordanous, “Voice Separation in Polyphonic Music: A Data-Driven Approach,” *Proceedings of the International Computer Music Conference*, Belfast, Ireland, 2008.

[8] Duane, B., & Pardo, B., “Streaming from MIDI using constraint satisfaction optimization and sequence alignment,” *Proceedings of the International Computer Music Conference*, 1–8, 2009.

[9] R. de Valk, T. Weyde, and E. Benetos, “A Machine Learning Approach to Voice Separation in Lute Tablature,” *Proceedings of the 14th International Society for Music Information Retrieval Conference*, pp 555–560, Curitiba, Brazil, 2013.

[10] McLeod, A. and Steedman, M., “A Machine Learning Approach to Voice Separation in Lute Tablature,” *Journal of New Music Research*, 45, 17-26, 2016.

[11] Patrick Gray, Razvan Bunescu. “A Neural Greedy Model for Voice Separation in Symbolic Music,” *17th International Society for Music Information Retrieval Conference*, 2016.

[12] Viterbi, A., “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *Information Theory, IEEE Transactions*, 13(2), 260–269, 1967.