A MACHINE LEARNING APPROACH TO VOICE SEPARATION IN LUTE TABLATURE

Reinier de Valk **Tillman Weyde** **Emmanouil Benetos**

Music Informatics Research Group
Department of Computer Science
City University London
{r.f.de.valk,t.e.weyde,Emmanouil.Benetos.1}@city.ac.uk

ABSTRACT

In this paper, we propose a machine learning model for voice separation in lute tablature. Lute tablature is a practical notation that reveals only very limited information about polyphonic structure. This has complicated research into the large surviving corpus of lute music, notated exclusively in tablature. A solution may be found in automatic transcription, of which voice separation is a necessary step. During the last decade, several methods for separating voices in symbolic polyphonic music formats have been developed. However, all but two of these methods adopt a rule-based approach; moreover, none of them is designed for tablature. Our method differs on both these points. First, rather than using fixed rules, we use a model that learns from data: a neural network that predicts voice assignments for notes. Second, our method is specifically designed for tablature—tablature information is included in the features used as input for the models—but it can also be applied to other music corpora. We have experimented on a dataset containing tablature pieces of different polyphonic textures, and compare the results against those obtained from a baseline hidden Markov model (HMM) model. Additionally, we have performed a preliminary comparison of the neural network model with several existing methods for voice separation on a small dataset. We have found that the neural network model performs clearly better than the baseline model, and competitively with the existing methods.

1. INTRODUCTION

The lute, an instrument widely used from the early sixteenth to the mid-eighteenth century, has left us with a considerable corpus of instrumental polyphonic music: over 860 print and manuscript sources survive, containing approximately 60,000 pieces [12]. This music is notated exclusively in lute tablature. Lute tablature is a practical notation that provides no direct pitch information and only limited rhythmic information, but instead instructs the player where to place the fingers on the fretboard and which strings to pluck (see Figure 1). It reveals very little about the polyphonic structure of the music it encodes, since it specifies neither to which polyphonic voice the tablature notes belong, nor what their individual durations are. Lute tablature's "alien nature" [5] is the principal reason why, apart from a number of specialist studies, this large and important corpus has so far escaped systematic musicological research.

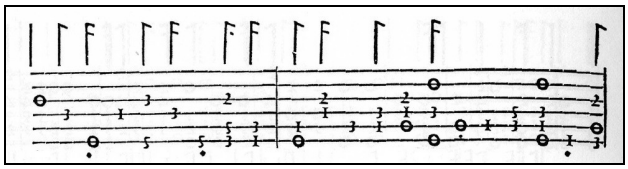

Figure 1. Excerpt of lute tablature in Italian style.

Transcription into modern music notation—a format much more familiar to the twenty-first-century scholar or musician—will increase the accessibility of the corpus, and, in fact, is the current modus operandi among those studying lute music. Transcribing tablature, however, is a time-consuming and specialist enterprise. Automatic transcription into modern music notation may provide a solution. An important step in the process of (automatic) transcription of polyphonic music is voice separation, i.e., the separation of the individual melodic lines ('voices') that together constitute the polyphonic fabric. Using machine learning techniques, we have developed two models for voice separation in lute tablature—a neural network model and a baseline hidden Markov model (HMM) model—which, with some modifications, can also be applied to other music corpora.

The outline of this paper is as follows: in Section 2, the existing methods for voice separation are discussed. In Section 3 the proposed models are introduced, and in Section 4 the dataset is presented. Section 5 is dedicated to the evaluation of the models; in Section 6 the results are discussed; and in Section 7 the performance of the neural network model is compared with that of several existing methods. Concluding thoughts are presented in Section 8.

2. RELATED WORK

During the last decade, several methods for separating voices in symbolic polyphonic music formats have been developed.[1] Except for two, described further below, all of these methods are rule-based. More concretely, they are based on at least one of two fundamental perceptual principles that group notes into voices, which have been

[1] In addition, a number of methods for voice separation in music in audio format exist—these, however, are left out of consideration here.

labelled the Pitch Proximity Principle and the Temporal Continuity Principle by Huron [6]. These principles imply that the closer notes are to one another in terms of pitch or time, respectively, the more likely they are perceived as belonging to the same voice. In addition, some of the methods include supplementary perceptual principles. Although these methods vary considerably in their approach, in each of them, the perceptual principles it is based on guide the voice assignment procedure.

Temperley [17] adopts an approach based on four 'preference rules,' i.e., criteria to evaluate a possible analysis. Two of these match the abovementioned principles; the other two prescribe to minimise the number of voices (New Stream Rule) and to avoid shared notes (Collision Rule). Cambouropoulos [1] briefly describes an elementary version of a voice separation algorithm based on the (Gestalt) principle of pitch proximity only. Chew and Wu [4] use a 'contig' approach, in which the music is divided into segments where a constant number of voices is active (the contigs). The voice fragments in the segments are then connected on the basis of pitch proximity; voice crossings are forbidden. Szeto and Wong [16] consider voices to be clusters containing events proximal in the pitch and time dimensions, and model voice separation as a clustering problem. The aim of their research, however, is to design a system for pattern matching, and not one for voice separation. In their method, voice separation is only a pre-processing step that prevents "perceptually insignificant stream-crossing patterns" from being returned by the system. Kilian and Hoos [9] present an algorithm that is not intended primarily for correct voice separation, but rather for creating "reasonable and flexible score notation." Their method allows for complete chords in a single voice. In the method presented by Karydis *et. al* [8], too, a 'voice' is not necessarily a "monophonic sequence of successive non-overlapping notes" [2]. Rather, they prefer to use the term 'stream,' which they define as "a perceptually independent voice consisting of single or multi-note sonorities." Hence, in addition to the 'horizontal' pitch and time proximity principles, they include two 'vertical integration' principles into their method: the Synchronous Note Principle (based on Huron's Onset Synchrony Principle) and the Principle of Tonal Fusion (based on Huron's Tonal Fusion Principle). A new version of this algorithm is described in Rafailidis et al. [14]. Madsen and Widmer [11], lastly, present an algorithm based primarily on the pitch proximity principle, with some heuristics added to handle unsolved situations.

In the remaining two methods, then, machine learning techniques are used. Kirlin and Utgoff [10] describe a system that consists of two components: a 'predicate,' implemented as a learned decision tree, that determines whether or not two notes belong to the same voice, and a hard-coded algorithm that then maps notes to voices. Jordanous [7] adopts a probabilistic approach based on a Markov model, and presents a system that learns the probability of each note belonging to each voice, as well as the probability of successive note pairs belonging to the same voice.

In addition to these more recent methods, another rule-based method—one designed specifically for automatic transcription of German lute tablature—was developed as early as the 1980s by Charnassé and Stepien [3]. In their research an approach was followed that combines expert knowledge encoded as rules with simpler heuristics. Although the results appear to be promising, the research seems to have ended prematurely.

3. PROPOSED MODELS

We have implemented two models for voice separation in tablature. The first uses a discrete hidden Markov model [13] to predict voice assignments for complete chords; the second uses a neural network (NN) to predict voice assignments for individual notes. The HMM model, in which the tablature chords are the only observations, is straightforward and functions as a baseline model to compare the neural network model against.

In our method, as in most existing methods, we use the notion of voice as a monophonic sequence of notes. In contrast to most rule-based methods, however, we allow voice crossings and shared notes (notes where two voices meet at the unison), both of which are perceptually problematic, but encountered frequently in polyphonic lute music. (This goes in particular for shared notes, which, especially in denser polyphony, are difficult to realise technically on the lute. Although actual unisons are sometimes used, a more idiomatic solution is to finger only one note of the unison—a technique also witnessed in keyboard music. Such notes shall henceforth be referred to as 'shared single notes.') Furthermore, unlike most existing methods, we assume in advance a maximum number of possible voices (five).[2]

3.1 HMM Model

We have used an HMM model in which the observations are the tablature chords, and the hidden states are the voice assignments. Each chord c is represented by a vector of pitches (MIDI numbers), depending on the number of notes in the chord ranging in length from 1 to 4; each voice assignment q_t for a given time frame t is represented by a vector of length 4. Here, each vector index represents a voice and can take the values -1, … , 3, where -1 denotes inactivity of the voice, and one of the other numbers the sequence number in the chord of the pitch that is assigned to that voice.

For each training set used in cross-validation, we have created a transition probability matrix $P(q_{t+1}|q_t)$, denoting the probability of having transitions between various voice assignments, an observation probability matrix $P(c_t|q_t)$, denoting the probability of encountering chord c_t given voice assignment q_t, and an initial state distribution $P(q_1)$. Since a training set might contain no instances of certain chord-voice assignment combinations, we have modified $P(c_t|q_t)$ by including a small non-zero

[2] Because of technical and physical limitations of lute and lutenist, more voices are rare in lute music.

probability for all cases where the number of pitches in a chord is the same as the number of assigned pitches in a voice assignment. This way, we discourage the prediction of voice assignments in which too few or too many pitches are assigned. Finally, the optimal voice assignment sequence is computed using the Viterbi algorithm [13].

It should be noted here that our HMM model is similar to Jordanous's system, as described in [7]. Firstly, both are probabilistic approaches, and second, only pitch-related observations from the training data are used. The main difference between her system and our HMM model is that in the former, a Markov chain with an ad-hoc cost function based on learned transition probabilities is used. Jordanous herself notes that "[i]t would be interesting to apply Hidden Markov Models . . . so that more of the previously allocated notes can be used to assist in voice allocation."

3.2 Neural Network Model

In the neural network model, the task of voice separation is modelled as a classification problem where every tablature note is assigned to a voice—or, in the case of a shared single note, to two voices. We used a standard feed-forward neural network with resilient backpropagation (Rprop) [15] and sigmoid activation function, which provides a proven fast and robust learning model.[3] The network consists of an input layer of 32 neurons, one hidden layer of 8 neurons, and an output layer of five neurons, each of which represents a voice. Having five output neurons enables us to use the network for five-voice lute music; however, because we are currently using a four-voice dataset, at the moment the fifth neuron is never activated. Using the sigmoid function, the individual output neurons all have activation values between 0 and 1; the neuron that gives the highest activation value determines the voice assignment decision. Prior to the actual training and testing, we have optimised the regularisation parameter λ (0.00003) and the number of hidden neurons (8) using a cross-validated grid search.

Using cross-validation and regularisation, we have trained in three runs, where each run consisted of 200 training epochs and the network weights were re-initialised randomly at the start of each run. The model from the training run in which the lowest error rate (see Section 5) was obtained, was selected for the validation stage.

In the validation stage, the model traverses the tablature note by note, from left to right (always starting with the lowest note in a chord), and assigns the notes to voices. The test process is linear, and previous voice assignments are not revised—except when an assignment conflict arises within a chord, i.e., when a note is assigned to a voice that was already assigned a note in the chord. Because we do not allow two notes within a chord to be assigned to the same voice, conflicts are solved using a heuristic that reassigns the current note to a yet unassigned voice. Since we have encountered only two conflicts in our experiments, we will not go into further details on this heuristic here. We assume that the low number of conflicts is due to the fact that the voices already assigned in the chord are given as a feature to the network (see next section).

3.2.1 Features

A 32-dimensional feature vector is generated for each tablature note, which contains two types of information (see Table 1). Features 1-12 contain only tablature information, and consist of (a) features encoding instrument-technical properties of the note (1-8), and (b) features encoding information about the position of the note within the chord (9-12). Features 13-32 contain information about the note's polyphonic embedding: (c) pitch and time proximities of the note to the previous note in each voice at the current onset time (13-27), and (d) the voices that have already been assigned to previous notes in the chord (28-32).

Three things should be noted here. First, features 13-27 encode, in essence, the principles that were labelled Pitch Proximity- and Temporal Continuity Principle by Huron [6]. Second, for the calculation of features 13-32, in addition to tablature information, voice assignment information is needed. Third, the time window within which the information is extracted that is used for the voice assignment decision, is presently still rather limited as it reaches back only one note per voice.

Tablature information	
Note information	7. isOrnamentation
1. pitch	8. isOpenCourse
2. course	**Chord information**
3. fret	9. numberOfNotesBelow
4. minDuration	10. numberOfNotesAbove
5. maxDuration	11. pitchDistanceToNoteBelow
6. chordSize	12. pitchDistanceToNoteAbove
Polyphonic embedding information	
Pitch/time proximities	23-27. offsetOnsetProx
13-17. pitchProx	**Voices already assigned**
18-22. interOnsetProx	28-32. voicesAlreadyAssigned

Table 1. Features for the NN model.

4. DATASET

At the moment, we are focusing on sixteenth-century lute music—more specifically, on intabulations, lute arrangements of polyphonic vocal pieces. There are three reasons for this choice. First, intabulations are highly representative of the entire sixteenth-century corpus since they then formed the predominant lute genre. Second, since the densest polyphonic structures in lute music are found in intabulations, they constitute a sub-corpus that is challenging for our research. Third, the use of intabulations provides an objective way of devising a ground truth by polyphonically aligning the tablature and the vocal pieces, whose voices are always notated separately. We have thus transcribed a number of carefully selected intabulations into modern music

[3] We use the implementation provided by the Encog framework. See http://www.heatonresearch.com/encog (accessed May 2013).

notation, and then converted these to MIDI, storing each voice in a separate file. The tablature encoding (in .txt format), together with the MIDI representation of the ground truth, are given as input to the model.

The dataset currently consists of nine intabulations, all for four voices (the most common intabulation format), and contains pieces of different polyphonic texture: three imitative pieces, three 'semi-imitative' pieces (pieces that contain points of imitation, but whose structure is not governed by them), and three free pieces. It comprises a total of 8892 notes divided over 5156 chords, single-note chords included (Table 2).

Piece	Texture	Notes	Chords
Ochsenkun 1558, *Absolon fili mi*	imitative	1184	727
Ochsenkun 1558, *In exitu Israel de Egipto*	imitative	1974	1296
Ochsenkun 1558, *Qui habitat*	imitative	2238	1443
Rotta 1546, *Bramo morir*	free	708	322
Phalèse 1547, *Tant que uiuray*	free	457	228
Ochsenkun1558, *Herr Gott laß dich erbarmen*	free	371	195
Abondante1548, *Mais mamignone*	semi-imitative	705	316
Phalèse 1563, *Las on peult*	semi-imitative	777	395
Barbetta 1582, *Il nest plaisir*	semi-imitative	478	234
Totals		8892	5156

Table 2. The dataset used for the experiments.

5. EVALUATION

5.1 Evaluation Metrics

Our main evaluation metric is the error rate, which is the percentage of notes assigned to an incorrect voice. The error rate is calculated by comparing, for each note, the predicted voice assignment with the ground truth voice assignment. For the NN model, we use two modes of evaluation. In *test* mode, we calculate the feature vectors with which the model is evaluated using the ground truth voice assignments. In *application* mode, which corresponds to the 'real-world situation' where the ground truth voice assignments are not provided, we calculate the feature vectors using the voice assignments predicted by the model. In application mode errors can propagate—once a note has been assigned to the wrong voice(s), this will influence the decision process for the assignment of the following notes or chords—typically resulting in higher error values. We thus distinguish between the *test error*, which is the error rate in test mode, and the *application error*, the error rate in application mode. For the HMM model, we evaluate using only a single metric that corresponds to the application error in the NN model.

Furthermore, for both models we use a tolerant and a strict approach for calculating errors—a distinction that applies to how shared single notes are handled. We distinguish between fully correct assignments (C), fully incorrect assignments (I) and three additional mixed categories: one voice assigned correctly but the other overlooked (O); one voice assigned correctly but another assigned superfluously (S); and one voice assigned correctly but the other assigned incorrectly (CI). All possibilities are listed in Table 3. In the tolerant evaluation approach, then, O, S, and CI are not counted as errors; in the strict approach they are counted as 0.5 errors.

P(*n*)	G(*n*)	Possibility	Error category				
			C	O	S	CI	I
1	1	P is G	X				
		P is not G					X
1	2	P is one of G		X			
		P is none of G					X
2	1	one of P is G			X		
		none of P is G					X
2	2	both P are G	X				
		one of P is G				X	
		none of P are G					X

Table 3. Error categories (P(*n*) = predicted voice(s) for note *n*; G(*n*) = ground truth voice(s) for note *n*).

5.2 Results

We have trained and evaluated both models on the complete dataset using nine-fold cross-validation, where the folds correspond to the individual pieces in the dataset and each piece serves as test set once. The results are given in Table 4.

Tolerant approach		Strict approach	
Error (%)	**Std. dev.**	**Error (%)**	**Std. dev.**
NN model, test error			
11.52	3.41	12.87	3.63
NN model, application error			
19.37	5.43	20.67	5.61
HMM model, application error			
24.95	6.59	25.64	6.69

Table 4. Averaged error rates (weighted) and standard deviation in cross-validation.

6. DISCUSSION

The performance of the models is compared by means of the application error rates. We see that the NN model outperforms the HMM model by about 5 percentage points—both when the tolerant and when the strict approach is applied. While the application error gives a realistic idea of how well the NN model actually performs, it is also interesting to have a look at the test error, which reflects the performance of the model when 'perfect' context information—context information derived directly from the ground truth voice assignments—is provided. A comparison of the test and application mode informs us about error propagation in the application mode. On the individual pieces, the test

errors are approximately between one half and two-thirds the size of the application errors, meaning that each misassigned note propagates 0.5-1.0 times. The high application errors might be explained at least partly by the observation that the pieces with high application errors contain many longer ornamental runs consisting of single notes, which are highly characteristic for lute music. Thus, when the first note of such a run is assigned to an incorrect voice, the following notes are very likely to be assigned to that voice as well. Because in such cases all notes are considered incorrect, single errors can propagate dramatically. However, the run as a whole will be assigned to a single voice, which is still a musically reasonable choice. This can be reflected using different evaluation metrics such as soundness and completeness (see Section 7).

We also observe that both models have problems handling shared single notes. In the NN model, 118 of the 129 shared single notes in the ground truth are assigned to only a single voice in test mode, and 114 in application mode. Moreover, 120 notes are superfluously assigned to a second voice in test mode, and 117 in application mode. We are currently using a simple heuristic to determine whether a note should be assigned to two voices: if the second highest activation value in the network output does not deviate more than 5.0% (the 'deviation threshold') from the highest activation value, the note is assigned to both corresponding voices. Although the current threshold leads to balanced results (118/114 shared single notes assigned erroneously to a single voice, versus 120/117 non-shared notes assigned superfluously to two), the method for determining shared single notes could be improved. In the HMM model, then, the number of shared single notes assigned erroneously to a single voice is in the same range (95); the number of notes assigned superfluously to two voices, however, is much lower (27). With respect to handling shared single notes, the HMM model overall thus performs better.

Voice crossings constitute another problem. An informal inspection shows that, in both models, most voice crossings are not detected. In the NN model, the main reason for this is that our features by design provide little support for voice crossings. This might be improved by including a 'melodic Gestalt criterion' in the form of features that represent melodic shape in the model. The inclusion of such features goes hand in hand with an increase of the information extraction window.

7. COMPARISON

We have compared our NN model with several of the existing methods for voice separation for which results and evaluation metrics are documented [4, 7, 10, 11, 14]. Using the same cross-validated procedure as above, but now excluding tablature-specific features such as course and fret, we have trained and tested the NN model on a small dataset that is comparable to those used in the above methods, and then evaluated the results using the different evaluation metrics proposed. It must be noted that the results of the comparison are only indicative, as the datasets used are similar but not identical and not all evaluation metrics are defined in detail.

Our dataset consists of the first five three-voice and the first five four-voice fugues of book I of Johann Sebastian Bach's *Wohltemperirtes Clavier*. [4] This collection of 48 preludes and fugues has been used, in total or in part, as the test set in most other methods we compare with—the only exception being the one described in [10], where the model is trained and tested on excerpts of the (stylistically comparable) chaconne from Bach's second violin partita (BWV 1004).

To enable a comparison we use five evaluation metrics: precision and recall, defined in [7] as "the percentage of notes allocated to a voice that correctly belong to that voice" (precision) and "the percentage of notes in the voice that are successfully allocated to that voice" (recall); soundness and completeness, defined in [10] as the percentage of adjacent note pairs in a predicted voice of which both notes belong to the same ground truth voice (soundness) and, conversely, the percentage of adjacent note pairs in a ground truth voice of which both notes have been assigned to the same predicted voice (completeness); and Average Voice Consistency (AVC) as used by [4], which measures, "on average, the proportion of notes from the same voice that have been assigned . . . to the same voice."

	Dataset	**Evaluation metric (%)**				
		P	R	S	C	A
NN	10 fugues (3-4vv)	83.12	83.12	94.07	93.42	82.67
[4]	48 fugues (3-5vv)					84.39
[7]	45 fugues (3-4vv)	80.88	80.85			
[10]	Bach chaconne			88.65	65.57	
[11]	30 Bach *Inventions* (2-3vv); 48 fugues (3-5vv)			95.94	70.11	
[14]	4 fugues (3-4vv)		92.50			

Table 5. Comparison of the NN model with other methods (P = precision; R = recall; S = soundness; C = completeness; A = Average Voice Consistency).[5]

As can be seen in Table 5, the results obtained by our NN model are in a similar range as those reported for the other models, and at times better. Moreover, with an application error of 16.87% (and a test error of 4.00%), the NN model performs better than on tablature (cf. Table 4).

[4] The dataset (in the form of MIDI files) was retrieved from www.musedata.org (accessed July 2013).

[5] In [11] it is stated that soundness and completeness "as suggested by Kirlin [and Utgoff]" were used as evaluation metrics; however, the textual definitions given differ. We have not yet been able to clarify this inconsistency, so we present the numbers and metrics exactly as in [11]. [14] use 'accuracy' as metric, whose definition matches that of recall.

8. CONCLUSIONS AND FUTURE WORK

In this paper we propose a neural network model for voice separation in lute tablature. This model is more flexible than the existing rule-based models in that it adapts to the data, and thus is less restricted with regard to what needs to be fixed as a priori rules. The model clearly outperforms the baseline HMM model and also seems to be more robust. In addition, it performs apparently competitively with the existing voice separation methods we have compared it with; however, extended tests will be needed for a systematic comparison. Although there is still room for improvement, the results are sufficiently promising to continue experimenting—not only with NN models, but also with different HMM models. Issues that need to be solved in particular are the high error propagation in the NN model's application mode, which currently complicates a real-world application, the handling of shared single notes, and the detection of voice crossings.

In future work, we will therefore extend the current NN model by including more features and by expanding the information extraction window. Additionally, we have started working on an approach that does not assign individual notes, but rather complete chords, to voices. With regard to the HMM model, we will experiment with more complex models using Gaussian mixture HMMs and factorial HMMs. Lastly, we are planning to work towards a more comprehensive and rigorous comparison of voice separation methods.

9. ACKNOWLEDGEMENTS

Reinier de Valk is supported by a City University London PhD Studentship and Emmanouil Benetos is supported by a City University London Research Fellowship.

10. REFERENCES

[1] E. Cambouropoulos: "From MIDI to Traditional Musical Notation," *Proceedings of the AAAI Workshop on Artificial Intelligence and Music*, n.p., 2000.

[2] E. Cambouropoulos: "'Voice' Separation: Theoretical, Perceptual and Computational Perspectives," *Proceedings of the 9th International Conference on Music Perception and Cognition*, pp. 987-997, 2006.

[3] H. Charnassé and B. Stepien: "Automatic Transcription of German Lute Tablatures: An Artificial Intelligence Application," *Computer Representations and Models in Music*, Ed. A. Marsden and A. Pople, Academic Press, London, pp. 144-70, 1992.

[4] E. Chew and X. Wu: "Separating Voices in Polyphonic Music: A Contig Mapping Approach," *Computer Music Modeling and Retrieval: Second International Symposium, Revised Papers*, Ed. U. K. Wiil, Springer, Berlin, pp. 1-20, 2004.

[5] J. Griffiths: "The Lute and the Polyphonist," *Studi Musicali*, Vol. 31, No. 1, pp. 89-108, 2002.

[6] D. Huron: "Tone and Voice: A Derivation of the Rules of Voice-Leading from Perceptual Principles," *Music Perception*, Vol. 19, No. 1, pp. 1-64, 2001.

[7] A. Jordanous: "Voice Separation in Polyphonic Music: A Data-Driven Approach," *Proceedings of the International Computer Music Conference*, n.p., 2008.

[8] I. Karydis et al.: "Horizontal and Vertical Integration/Segregation in Auditory Streaming: A Voice Separation Algorithm for Symbolic Musical Data," *Proceedings of the 4th Sound and Music Computing Conference*, pp. 299-306, 2007.

[9] J. Kilian and H. Hoos: "Voice separation—A Local Optimisation Approach," *Proceedings of the 3rd International Conference on Music Information Retrieval*, n.p., 2002.

[10] P. Kirlin and P. Utgoff: "VoiSe: Learning to Segregate Voices in Explicit and Implicit Polyphony," *Proceedings of the 6th International Conference on Music Information Retrieval*, pp. 552-557, 2005.

[11] S. T. Madsen and G. Widmer: "Separating Voices in MIDI," *Proceedings of the 7th International Conference on Music Information Retrieval*, n.p., 2006.

[12] A. J. Ness and C. A. Kolczynski: "Sources of Lute Music," *The New Grove Dictionary of Music and Musicians*, 2nd ed., Ed. S. Sadie, Macmillan, London, pp. 39-63, 2001.

[13] L. R. 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.

[14] D. Rafailidis, E. Cambouropoulos, and Y. Manolopoulos: "Musical Voice Integration/ Segregation: *VISA* Revisited," *Proceedings of the 6th Sound and Music Computing Conference*, pp. 42-47, 2009.

[15] M. Riedmiller and H. Braun: "RPROP—A Fast Adaptive Learning Algorithm," *Proceedings of the International Symposium on Computer and Information Science*, n.p., 1992.

[16] W. M. Szeto and M. H. Wong: "Stream Segregation Algorithm for Pattern Matching in Polyphonic Music Databases," *Multimedia Tools and Applications*, Vol. 30, pp. 109-127, 2006.

[17] D. Temperley: *The Cognition of Basic Musical Structures*, The MIT Press, Cambridge, MA, 2001.