L^p-VITERBI ALGORITHM FOR AUTOMATIC FINGERING DECISION

Gen Hori
Asia University / RIKEN
hori@brain.riken.jp

Shigeki Sagayama
Meiji University
sagayama@meiji.ac.jp

ABSTRACT

A theoretical basis of existing attempts to automatic fingering decision is path optimization that minimizes the difficulty of whole phrase that is typically defined as the sum of the difficulties of each moves required for playing the phrase. However, from a practical point of view, it is more important to minimize the maximum difficulty of the move required for playing the phrase, that is, to make the most difficult move easiest. For this reason, our previous work introduced a variant of the Viterbi algorithm termed the "minimax Viterbi algorithm" that finds the path of the hidden states that maximizes the minimum transition probability along the path. In the present work, we introduce a parameterized family of Viterbi algorithm termed the "L^p-Viterbi algorithm" that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm. (It coincides with the conventional for $p = 1$ and the minimax for $p = \infty$.) We apply those variants of the Viterbi algorithm to HMM-based guitar fingering decision and compare the resulting fingerings.

1. INTRODUCTION

The pitch ranges of strings of string instruments usually have significant overlaps. Therefore, such string instruments have several ways to play even a single note (except the highest and the lowest notes that are covered only by a single string) and thus numerous ways to play a phrase and an astronomical number of possible ways to play a whole song. This is why fingering decision for a given song is not an easy task and automatic fingering decision has been attracting many researchers. Existing attempts to automatic fingering decision are mainly based on path optimization by minimizing the difficulty level of whole phrase that is defined as the sum or the product of the difficulty levels of each moves. However, whether a string player can play a given passage using a specific fingering depends almost only on whether the most difficult move included in the fingering is playable. In particular, for beginner players, it is most important to minimize the maximum difficulty level of move included in a fingering, that is, to "make the most difficult move easiest."

To this end, our previous work [1] introduced a variant of the Viterbi algorithm [2] termed the "minimax Viterbi algorithm" for finding the sequence of the hidden states that maximizes the minimum transition probability on the sequence (not the product of all the transition probabilities on the sequence). The purpose of the paper is to introduce a parameterized family of Viterbi algorithm termed the "L^p-Viterbi algorithm" that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm and then apply it to HMM-based guitar fingering decision. The framework of HMM-based fingering decision employs a hidden Markov model (HMM) whose hidden states are left hand forms of guitarists and output symbols are musical notes. We perform fingering decision by solving the decoding problem of HMM using our proposed L^p-Viterbi algorithm. We set the transition probabilities to large values for easy moves and small values for difficult ones so that resulting fingerings make the most difficult move easiest as previously discussed in this section. To distinguish the original Viterbi algorithm from our variants, we refer to the former as "conventional Viterbi algorithm" throughout the paper.

There have been several attempts to automatic fingering decision and automatic arrangement for guitars. Sayegh [3] introduced the path optimization approach to fingering decision of generic string instruments. Miura et al. [4] developed a system that generates guitar fingerings for given melodies (monophonic phrases). Radicioni et al. [5] introduced cognitive aspects of fingering decision to the path optimization approach. Radisavljevic and Driessen [6] proposed a method for designing cost functions for dynamic programming (DP) for fingering decision. Tuohy and Potter [7] introduced a genetic algorithm (GA) for fingering decision. Baccherini et al. [8] introduced finite state automaton to fingering decision of generic string instruments. Hori et al. [9] applied input-output HMM introduced by Bengio and Frasconi [10] to guitar fingering decision and arrangement. McVicar et al. [11] developed a system that generates guitar tablatures for rhythm guitar and lead guitar automatically. Comparing to those previous works, the novelty of the present work lies in that it introduces a new category of path optimization and variants of the Viterbi algorithm correspondingly.

The rest of the paper is organized as follows. Section 2 recalls the conventional Viterbi algorithm and the minimax Viterbi algorithm and connects them by introducing the L^p-Viterbi algorithm. Section 3 recalls the framework of fingering decision based on HMM for monophonic guitar phrases to which we apply the L^p-Viterbi algorithm and evaluates the resulting fingerings in Section 4. Section 5 concludes the paper.

2. L^P-VITERBI ALGORITHM

First of all, we recall the definition of HMM and the procedure of the conventional Viterbi algorithm for finding the sequence of hidden states with the maximum likelihood. Next, we look at the minimax Viterbi algorithm introduced in our previous work [1] for finding the sequence of hidden states with the maximum minimum transition probability. After that, we introduce the L^p-Viterbi algorithm that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm.

2.1 Hidden Markov model (HMM)

Suppose that we have two finite sets of hidden states Q and output symbols S,

$$Q = \{q_1, q_2, \ldots, q_N\},$$
$$S = \{s_1, s_2, \ldots, s_M\},$$

and two sequences of random variables $\boldsymbol{X}$ of hidden states and $\boldsymbol{Y}$ of output symbols,

$$\boldsymbol{X} = (X_1, X_2, \ldots, X_T), \quad X_t \in Q,$$
$$\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_T), \quad Y_t \in S,$$

then a hidden Markov model M is defined by a triplet

$$M = (A, B, \pi)$$

where A is an $N \times N$ matrix of the transition probabilities,

$$A = (a_{ij}),\ a_{ij} \equiv a(q_i, q_j) \equiv P(X_t = q_j | X_{t-1} = q_i),$$

B an $N \times K$ matrix of the output probabilities,

$$B = (b_{ik}),\ b_{ik} \equiv b(q_i, s_k) \equiv P(Y_t = s_k | X_t = q_i),$$

and Π an N-dimensional vector of the initial distribution of hidden states,

$$\Pi = (\pi_i),\ \pi_i \equiv \pi(q_i) \equiv P(X_1 = q_i).$$

2.2 Conventional Viterbi algorithm

When we observe a sequence of output symbols

$$\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$$

from a hidden Markov model M, we are interested in the sequence of hidden states

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_T)$$

that generated the observed sequence of output symbols $\boldsymbol{y}$ with the maximum likelihood,

$$\begin{aligned}
\hat{\boldsymbol{x}}_{ML} &= \arg\max_{\boldsymbol{x}} P(\boldsymbol{y}, \boldsymbol{x} | M) \\
&= \arg\max_{\boldsymbol{x}} P(\boldsymbol{x}|M) P(\boldsymbol{y}|\boldsymbol{x}, M) \\
&= \arg\max_{\boldsymbol{x}} (\log P(\boldsymbol{x}|M) + \log P(\boldsymbol{y}|\boldsymbol{x}, M)) \\
&= \arg\max_{\boldsymbol{x}} \sum_{t=1}^{T} (\log a(x_{t-1}, x_t) + \log b(x_t, y_t))
\end{aligned} \tag{1}$$

where we write $\pi(x_1) = a(x_0, x_1)$ for convenience. The problem of finding the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ is called the decoding problem and solved efficiently using two $N \times T$ tables $\Delta = (\delta_{it})$ of maximum log likelihood and $\Psi = (\psi_{it})$ of back pointers and the following four steps.

Initialization initializes the first columns of the two tables Δ and Ψ using the following formulae for $i = 1, 2, \ldots, N$,

$$\delta_{i1} = \log \pi_i + \log b(q_i, y_1),$$
$$\psi_{i1} = 0.$$

Recursion fills out the rest columns of Δ and Ψ using the following recursive formulae for $j = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$–1,

$$\delta_{j,t+1} = \max_i (\delta_{it} + \log a_{ij}) + \log b(q_j, y_{t+1}),$$
$$\psi_{j,t+1} = \arg\max_i (\delta_{it} + \log a_{ij}).$$

Termination finds the last hidden state of the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ using the last column of Δ,

$$x_T = \arg\max_i \delta_{iT}.$$

Backtracking tracks the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ from the last to the first using the back pointers of Ψ for $t = T, T$–$1, \ldots, 2$,

$$x_{t-1} = \psi_{x_t, t}.$$

2.3 Minimax Viterbi algorithm

To implement a variant of the conventional Viterbi algorithm for finding the sequence of hidden states $\hat{\boldsymbol{x}}_{MM}$ with the maximum minimum transition probability,

$$\hat{\boldsymbol{x}}_{MM} = \arg\max_{\boldsymbol{x}} \min_t (\log a(x_{t-1}, x_t) + \log b(x_t, y_t)) \tag{2}$$

where the minimum value is taken from $t = 1, 2, \ldots, T$, we modify the second step of the conventional Viterbi algorithm as follows.

Recursion for minimax Viterbi algorithm fills out the two tables Δ and Ψ using the following recursive formulae for $j = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$–1,

$$\delta_{j,t+1} = \max_i (\min(\delta_{it},\ \log a_{ij} + \log b(q_j, y_{t+1}))),$$
$$\psi_{j,t+1} = \arg\max_i (\min(\delta_{it},\ \log a_{ij} + \log b(q_j, y_{t+1}))).$$

We modify only the second step and leave other steps unchanged. We call the modified algorithm the "minimax Viterbi algorithm." The modified algorithm finds the sequence of hidden states with the maximum minimum probability, that is, the minimum maximum difficulty. Although the word "maximin" is appropriate in view of probability, we choose the word "minimax" for naming the variant because it is intuitive to understand in view of difficulty and the word conveys our concept of "making the most difficult

move easiest." The modified second step works in the same manner as the original one but now the element δ_{it} keeps the value of the maximum minimum transition probability of the subsequence of hidden states for the first t observations. The term $\min(\delta_{it}, \log a_{ij} + \log b(q_j, y_{t+1}))$ updates the value of the minimum transition probability as the term $\delta_{it} + \log a_{ij} + \log b(q_j, y_{t+1})$ does the value of the maximum log likelihood in the conventional Viterbi algorithm.

2.4 L^p-Viterbi algorithm

To implement a family of Viterbi algorithm that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm, we consider a generalized decoding problem of HMM defined with the L^p-norm of a certain real vector. For a real number $p \geq 1$ [1], the L^p-norm of an n-dimensional real vector

$$\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in R^n$$

is defined as

$$||\boldsymbol{v}||_p = \sqrt[p]{|v_1|^p + |v_2|^p + \cdots + |v_n|^p}. \tag{3}$$

In particular, the L^1-norm and the L^2-norm correspond to the Manhattan distance and the Euclidean distance and are widely used in the contexts of machine learning and compressed sensing. The regression models with the L^1 and L^2 regularizations are the LASSO regression and the ridge regression respectively. The L^∞-norm (the limit of the L^p-norms for $p \to \infty$) is equivalent to the following definition,

$$||\boldsymbol{v}||_\infty = \max\{|v_1|, |v_2|, \ldots, |v_n|\}.$$

We define two T-dimensional real vectors of the transition probabilities $\boldsymbol{a}$ and the output probabilities $\boldsymbol{b}$ along the sequences of hidden states $\boldsymbol{x}$ and output symbols $\boldsymbol{y}$,

$$\boldsymbol{a}(\boldsymbol{x}) = (\pi(x_1), a(x_1, x_2), \ldots, a(x_{T-1}, x_T)),$$
$$\boldsymbol{b}(\boldsymbol{x}, \boldsymbol{y}) = (b(x_1, y_1), b(x_2, y_2), \ldots, b(x_T, y_T)),$$

and consider a generalized decoding problem,

$$\hat{\boldsymbol{x}}_{L^p} = \arg\min_{\boldsymbol{x}} ||\log \boldsymbol{a}(\boldsymbol{x}) + \log \boldsymbol{b}(\boldsymbol{x}, \boldsymbol{y})||_p, \tag{4}$$

where $\log$ operates element-wise on vectors. The special cases of $p=1$ and $p=\infty$ of the generalized decoding problem (4) coincide with the conventional decoding problem (1) and the minimax decoding problem (2) respectively. Thus the generalized decoding problem continuously interpolates the conventional and the minimax decoding problems. Note that $\arg\max$ in (1) and (2) changes to $\arg\min$ in (4) because the logarithms of probabilities are always negative or zero and therefore taking the absolute values of them in the L^p-norm (3) always inverts their signs. All the elements of the table Δ are negative or zero in the conventional and the minimax Viterbi algorithm while they are all positive or zero in the generalized decoding problem. We call the generalized decoding problem (4) the "L^p-decoding problem." To solve the L^p-decoding problem efficiently, we modify the second step of the conventional Viterbi algorithm as follows. According to the change of signs in the table Δ, we also modify the first and third step as follows. We leave the last step unchanged.

Initialization for L^p-Viterbi algorithm initializes the first columns of the two tables Δ and Ψ using the following formulae for $i = 1, 2, \ldots, N$,

$$\delta_{i1} = |\log \pi_i + \log b(q_i, y_1)|,$$
$$\psi_{i1} = 0.$$

Recursion for L^p-Viterbi algorithm fills out the two tables Δ and Ψ using the following recursive formulae for $j = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$–1,

$$\delta_{j,t+1} = \min_i \sqrt[p]{{\delta_{it}}^p + |\log a_{ij} + \log b(q_j, y_{t+1})|^p},$$
$$\psi_{j,t+1} = \arg\min_i \sqrt[p]{{\delta_{it}}^p + |\log a_{ij} + \log b(q_j, y_{t+1})|^p}.$$

Termination for L^p-Viterbi algorithm finds the last hidden state of the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ using the last column of Δ,

$$x_T = \arg\min_i \delta_{iT}.$$

3. FINGERING DECISION BASED ON HMM

We evaluate our proposed L^p-Viterbi algorithm in the following section with the same fingering decision model for monophonic guitar phrases used in our previous work [1]. In this section, we look at the monophonic fingering decision model based on HMM whose hidden states are left hand forms and output symbols are musical notes played by the left hand forms. In this formulation, fingering decision is cast as a decoding problem of HMM where a fingering is obtained as a sequence of hidden states.

3.1 HMM for monophonic fingering decision

To play a single note with a guitar, a guitarist depresses a string on a fret with a finger of the left hand and picks the same string with the right hand. Therefore, a form q_i for playing a single note can be expressed in a triplet

$$q_i = (s_i, f_i, h_i)$$

where $s_i = 1, \ldots, 6$ is a string number (from the highest to the lowest), $f_i = 0, 1, \ldots$ is a fret number, and $h_i = 1, 2, 3, 4$ is a finger number of the player's left hand (1,2,3 and 4 means the index, middle, ring and pinky fingers). The fret number $f_i = 0$ means an open string. For the standard tuning (E_4-B_3-G_3-D_3-A_2-E_2), the MIDI note numbers of the open strings are $o_1 = 64, o_2 = 59$, $o_3 = 55, o_4 = 50, o_5 = 45, o_6 = 40$ from which the MIDI note number of the note played by the form q_i is calculated as

$$n(q_i) = o_{s_i} + f_i.$$

[1] For $p < 1$, $||\boldsymbol{v}||_p$ does not meet the axioms of norm.

3.2 Transition and output probabilities

The model parameters of HMM such as the transition probabilities and the output probabilities are usually estimated from training data using the Baum-Welch algorithm [12]. However, we set those parameters manually as explained in the following instead of estimation from training data because it is difficult to prepare enough training data for our application of fingering decision, that is, guitar scores annotated with fingerings.

The difficulty levels of the moves from forms to forms are implemented in the probabilities of the transitions from hidden states to hidden states; a small value of the transition probability means the corresponding move is difficult and a large value means easy. For simplicity, we assume that the four fingers of the left hand (the index, middle, ring and pinky fingers) are always put on consecutive frets. This lets us calculate the *index finger position* (the fret number the index finger is put on) of form q_i as follows,

$$g(q_i) = f_i - h_i + 1.$$

Using the index finger position, we set the transition probability as

$$\begin{aligned} a_{ij}(d_t) &= P(X_t = q_j \,|\, X_{t-1} = q_i,\, d_t) \\ &\propto \frac{1}{2d_t} \exp\left(-\frac{|g(q_i) - g(q_j)|}{d_t}\right) \times P_H(h_j) \end{aligned}$$

where $\propto$ means proportional and the left hand side is normalized so that the summation with respect to j equals 1. The first term of the right hand side is taken from the probability density function of the Laplace distribution that concentrates on the center and its variance d_t is set to the time interval between the onsets of the $(t-1)$-th note and the t-th note. The second term $P_H(h_j)$ corresponds to the difficulty level of the destination form defined depending on the finger number h_j. In the simulation in the following section, we set $P_H(1) = 0.4$, $P_H(2) = 0.3$, $P_H(3) = 0.2$ and $P_H(4) = 0.1$ which means the form using the index finger is the easiest and the pinky finger is the most difficult. The difficulty levels of the forms are included in the transition probabilities (not in the output probability) in such a way that a transition probability is adjusted to a smaller value when the destination form of the transition is difficult.

As for the output probability, because all the hidden states have unique output symbols in our HMM for fingering decision, it is 1 if the given output symbol is the one that the hidden state outputs and 0 if the given output symbol is not,

$$\begin{aligned} b_{ik} &= P(\, Y_t = s_k \,|\, X_t = q_i \,) \\ &= \begin{cases} 1 & (\text{if } s_k = n(q_i)) \\ 0 & (\text{if } s_k \neq n(q_i)) \end{cases} . \end{aligned}$$

4. EVALUATION

We evaluate our proposed L^p-Viterbi algorithm by comparing the results of fingering decision for monophonic guitar phrases using the conventional Viterbi algorithm, the minimax Viterbi algorithm, and the proposed algorithm. Figure 1 shows the results for the opening part of "Romance Anonimo." The numbers on the tablatures show the fret numbers and the numbers in parenthesis below the tablatures show the finger numbers where 1,2,3 and 4 means the index, middle, ring and pinky fingers. In the bottom graph of the transition probabilities, we see that the red line (the conventional Viterbi algorithm) keeps higher values (easy moves) at the cost of two very small values (very difficult moves) while the blue line (the minimax Viterbi algorithm) circumvents such very small values and makes the most difficult move easiest. The green line (the L^p-Viterbi algorithm for $p = 3.0$) falls into an intermediate category. The top tablature uses the pinky finger two times while the middle and the bottom ones avoids it. Figure 2 shows the results for an excerpt from the cello part of "Eine Kleine Nachtmusik." In the bottom graph, we see again that the red line keeps higher values at the cost of few very small values while the blue line circumvents such very small values.

5. CONCLUSIONS

We have introduced a variant of the conventional Viterbi algorithm termed the L^p-Viterbi algorithm that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm. The variant coincides with the conventional Viterbi algorithm for $p = 1$ and the minimax Viterbi algorithm for $p = \infty$. We have applied the variant to HMM-based guitar fingering decision. Our previous work [1] showed that the minimax Viterbi algorithm minimizes the maximum difficulty of the move required for playing a given phrase. In the simulation of the present work, we have compared the fingerings obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm and the minimax Viterbi algorithm to see that the L^p-Viterbi algorithm is capable of making fingerings that fall into an intermediate category between the conventional Viterbi algorithm and the minimax Viterbi algorithm. We hope that those variants of the Viterbi algorithm find a wide range of applications in a variety of research fields.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers 26240025, 17H00749.

6. REFERENCES

[1] G. Hori and S. Sagayama, "Minimax viterbi algorithm for HMM-based guitar fingering decision," in *Proceedings of 17th International Society for Music Information Retrieval (ISMIR2016)*, New York City, U.S.A., 2016, pp. 448–453.

[2] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[3] S. I. Sayegh, "Fingering for string instruments with

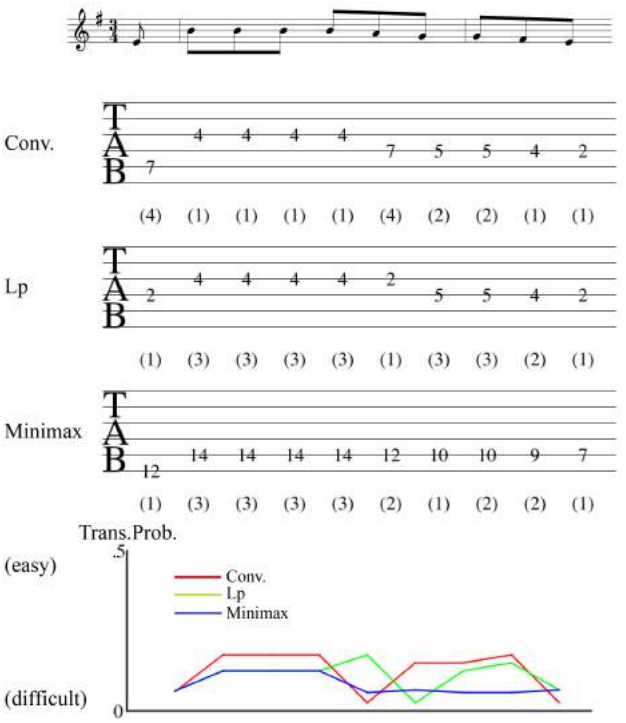

Figure 1. The results of fingering decision for the opening part of "Romance Anonimo" (only top notes). The top, the middle and the bottom tablatures show three fingerings obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm ($p = 3.0$) and the minimax Viterbi algorithm respectively. In the bottom graph, the red line shows the transition probabilities along the hidden path obtained by the conventional Viterbi algorithm while the green line and the blue line are obtained by the L^p-Viterbi algorithm and the minimax Viterbi algorithm respectively.

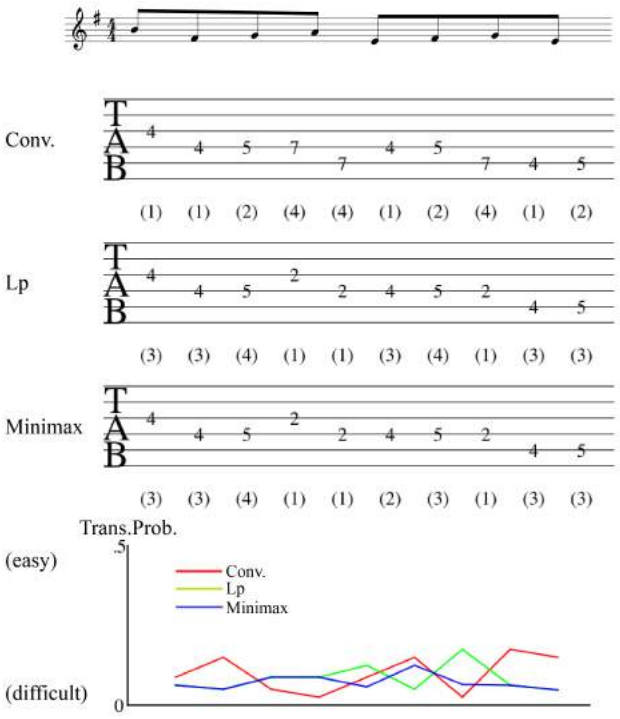

Figure 2. The results of fingering decision for an excerpt from the cello part of "Eine Kleine Nachtmusik." The top, the middle and the bottom tablatures show three fingerings obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm ($p = 3.0$) and the minimax Viterbi algorithm respectively. In the bottom graph, the red line, the green line and the blue line are obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm and the minimax Viterbi algorithm respectively.

the optimum path paradigm," *Computer Music Journal*, vol. 13, no. 3, pp. 76–84, 1989.

[4] M. Miura, I. Hirota, N. Hama, and M. Yanagida, "Constructing a system for finger-position determination and tablature generation for playing melodies on guitars," *Systems and Computers in Japan*, vol. 35, no. 6, pp. 10–19, 2004.

[5] D. P. Radicioni, L. Anselma, and V. Lombardo, "A segmentation-based prototype to compute string instruments fingering," in *Proceedings of Conference on Interdisciplinary Musicology (CIM04)*, vol. 17, Graz, Austria, 2004, pp. 97–104.

[6] A. Radisavljevic and P. Driessen, "Path difference learning for guitar fingering problem," in *Proceedings of International Computer Music Conference*, vol. 28, Miami, USA, 2004.

[7] D. R. Tuohy and W. D. Potter, "A genetic algorithm for the automatic generation of playable guitar tablature," in *Proceedings of International Computer Music Conference*, 2005, pp. 499–502.

[8] D. Baccherini, D. Merlini, and R. Sprugnoli, "Tablatures for stringed instruments and generating functions," in *Fun with Algorithms*. Springer, 2007, pp. 40–52.

[9] 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.

[10] Y. Bengio and P. Frasconi, "An input output HMM architecture," *Advances in neural information processing systems*, vol. 7, pp. 427–434, 1995.

[11] M. McVicar, S. Fukayama, and M. Goto, "Autoleadguitar: Automatic generation of guitar solo phrases in the tablature space," in *12th International Conference on Signal Processing (ICSP)*, 2014, pp. 599–604.

[12] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *The annals of mathematical statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.