コード進行 HMM を用いた伴奏生成支援システムの構築

小野 まなつ
Manatsu Ono
法政大学大学院情報科学部ディジタルメディア学科
manatsu.ono.4v@stu.hosei.ac.jp

Abstruct

There is a system that creates accompaniment using a automatic arrangement function, but its completeness is not enough. In this research, a chord progression is created that is closer to real songs by learning existing songs. Output chords are chosen based on likelihoods of histgrams of sounds constituting a melody for each chords. Next, likelihoods of chord transition are calculated using ergodic HMM to increase the accuracy of transition than before. The final output is a chord progression with a higher by multiplying these likelihoods. As a result of comparing with existing songs, the degree of reproduction was 72.2%, which was an improvement of 17.9% over the existing system. From the above, the accuracy of chord selection is improved more than the conventional method, and it possible to create more appropriate chord progression.

1 はじめに

従来の伴奏生成システムとして Band In A Box、MySong[1] の 2 つが挙げられる。Band In A Box には自動作曲や自動編曲の機能が備わっており、その中の 1 つとしてメロディからコードを提案するという機能がある。ここでのコードというのは、メロディに合った和音ではあるが、コード進行は考慮されておらず、曲を通して聞いた時に不自然な伴奏になってしまう事がある。対して MySong は既存の曲をデータベースとして学習しており、HMM[2] を利用して和音を選択しているためコード進行も考慮された伴奏となる。しかし、Band in A Box、MySong どちらについても、既存の曲で伴奏を付けた時のもとのコードの再現度はあまり高くなく、メロディに対して合わないコードが付与されてしまうことも度々ある。そこで本研究では MySong と同様に HMM を利用した伴奏生成機能を作成し、その精度の向上を目標とする。

2 メロディに対するコードの選択方法

2.1 確率モデルによるコード選択

あるメロディにコード進行を付与する事を考えた時、同じメロディであっても当てはめる事の出来るコードはいくつかあるが、一般的にコードの構成音とメロディは合うものが組み合わされている必要がある。そこで、既存の曲についてメロディがどのような音で構成されており、そのときにどのようなコードを当てる事が出来るのかを分布に起こし、コードを付けたいメロディの音の分布の場合にどのコードを当てる事が出来るのかを確率モデルによって求めていく。また、いくつかのコードを繋げていく場合、和声法など前後の繋がりも考慮する必要があり、ただコードがメロディに合っていれば良いという訳ではない。そこで今回は既存の曲のコード進行を HMM で学習し、一般的なコードの繋げ方に近いものを選んでいく方法をとる。

2.2 メロディの音に合うコードの選択

使用可能なコードの候補をしぼっていくために、メロディの音の分布を多項分布としてデータベースとの尤度を求める。入力されたメロディ 1 音ずつについて、その音がどのコードが出てきたと考えるのが尤もらしいか、というのが今回用いる尤度である。メロディの音が C だった場合、データベースの全てのコードの C のビンの割合が尤度となり、メロディ全てについてこの尤度を求める。尤度は値が小さくなりすぎないように対数をとって計算していく。

2.3 Erogodic HMM

メロディに含まれる音階を基に、演奏可能なコードを選択していく。伴奏としてコードを並べていくために、ここで Ergodic HMM[3] を利用する。

HMM というのは時間と共に確率分布が変化する確率変数の系列のモデルであり、現在の状態から次の状態への遷移を考えるのに有効である。HMM には Left-to-Right HMM と Ergodic HMM の 2 種類がある。 Left-to-Right HMM では、状態を選んでいくときに必ず左から右へ遷移していくため、次の状態に遷移すると前の状態に戻ることはできない。対して Ergodic HMM は任意の状態から全ての状態に遷移するこができ、次の状態に遷移しても前の状態に戻ることが可能である。今回は曲の中で用いられているコードの繰り返し構造をカバーできる Ergodic HMM を利用する。Ergodic HMM では状態数が増えるほど構造が複雑になり、初めのうちは状態数が多くなるにつれて精度も上がっていくが、ある部分を境に複雑になりすぎて精度が下がるポイントが存在するはずである。

Ergodic HMM において、$p(v_k|i)$ を状態 i から遷移する際に記号 v_k を出力する確率と定義して、HMM(λ) のエントロピーについて、
状態 i でシンボル v_k を生成する確率は

$$p(v_k|i) = \sum_{j=1}^{N} a_i j b_i j(v_k) \tag{1}$$

1 シンボル当たりのエントロピーは

$$H(K|i) = -\sum_{k=1}^{K} p(v_k|i) log_2 p(v_k|i) \tag{2}$$

モデル λ のエントロピーは

$$H(\lambda) = \sum_{i=1}^{N} \omega_i H(K|i) \tag{3}$$

と求めることが出来る。但し、

N…HMM の状態数
K…シンボルの数 (種類)
$a_i j$…状態 i から状態 j へ遷移する確率
ω_i…状態 i の定常状態確率

である。Ergodic HMM において、状態 S_i から遷移を開始し、n 回の遷移を繰り返した後に状態 S_i に達する確率 (n 次の遷移確率) を $a_i j^{(n)}$ と表すことにする。$a_i j^{(n)}$ には次の式が成立する。

$$a_i j^{(n+1)} = \sum_{v=1}^{N} a_i v a_b j^{(n)} \tag{4}$$

$$a_i j^{(n+m)} = \sum_{v=1}^{N} a_i v^{(n)} a_b j^{(m)} \tag{5}$$

Supervisor: Prof. Katunobu Itou

n が大きくなるにつれて、$a_i j^{(n)}$ は一定値に近づき、その値は状態 S_i のみで決まり、出発点 S_i には無関係になる事が証明できる。すなわち、

$$\lim_{n\to\infty} a_i j^{(n)} = \omega_j \tag{6}$$

となる。ω_i は、十分な遷移の後において、任意の瞬間にこの過程が状態 S_i にある確率を表す。定常状態確率 ω_j には次の式が成り立つ。

$$\sum_{j=1}^{N} \omega_i = 1 \sum_{j=1}^{N} \omega_i a_i j = \omega_j \tag{7}$$

したがって、定常状態確率 ω_j は遷移確率から求められる。

例として、 状態 0、 状態 1 に対して出力記号 x と y があったとする。 この HMM が「xxy」という列を生成する確率を求めるには、 まず HMM が「xxy」を出力する全ての経路を列挙し、 各経路を通る確率を求める。 それらすべての経路の確率の和が、 HMM 今回は「xxy」を生成する確率になる。図 1 より、xxy の生成確率を合計すると 0.0752 になる。以上の手順で求められた尤度の対数値の和を求め、平均をとったものを平均尤度とする。

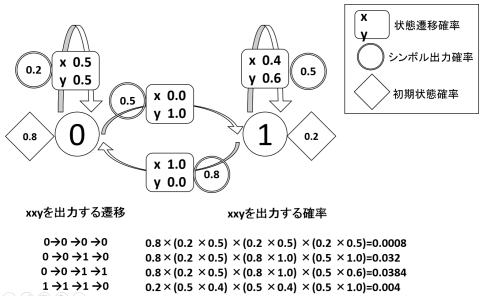

図 1　尤度の計算方法

今回は図 2 のような 5 状態のトポロジーの ErgodicHMM を利用する。HMM には初期状態の状態出力確率と各状態への遷移確率を与えておく。状態出力確率はそれぞれの状態の時にコードが出力される確率であり、初期値は合計が 1 となるような乱数で与える。トポロジーとなる遷移確率は、同じ状態に帰ってくる確率、開始状態から直接終了状態へ行く確率、終了状態から他の状態へ行く確率は 0 としておく。この初期状態を基にデータベース 2 からコード遷移を学習し、学習後の状態出力確率と遷移確率を用いてコードの尤度を求めていく。

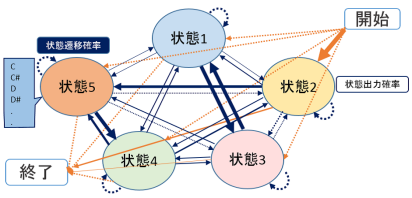

図 2　5 状態の場合の Ergodic HMM

2.4　出力コードの候補

まず、コードの選択のために HMM を学習する。コード遷移についてのデータベースを学習すると、各状態でのコードの出力確率と、状態間での遷移確率が求められる。続いて、入力されたメロディに対する各コードの出力尤度を求める。データベース 1 では今回利用するコード全てについて、そのコードが演奏されている時にメロディに含まれている音階の割合が求められている。1 小節分（又はコードの切り替えを行う拍数分）のメロディの音階それぞれと全コードについての対数尤度を足し合わせたものが、その小節でのコードの出力尤度となる。

1 小節毎にコードの尤度を出すために、各状態の出力確率にその小節のメロディに対する出力尤度を同じコード同士で足し合わせる。この際、メロディに対して合うコードが出力されることが前提なので、メロディに対するコードの出力尤度を 0.8 倍、各状態での出力確率を 0.2 倍にしている。各状態で 1 番尤度が高いコードを選び、前の各状態からの遷移確率との和を取るとメロディに対する出力尤度、学習した HMM での出力確率と遷移確率全てを考慮した尤度を求める事が出来る。ここで図 3 のように状態数の 2 乗の数コードの候補が出てくるため、さらにその中で尤度が最大のものを出力する。

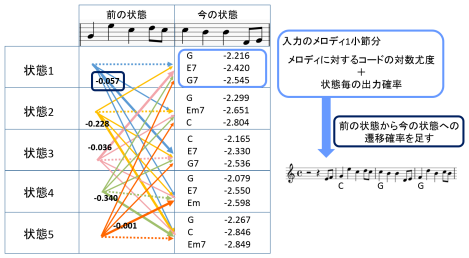

図 3　ErgodicHMM を用いたコードの尤度の求め方

各小節についてこの処理を行い、繋げたものがメロディ 1 曲分のコード進行となる。結果はコード名とその小節番号を出力する。コードの修正を行いたい場合はユーザーが小節番号を指定することにより、各状態について出力確率が高いコードを 10 個ずつ、重複しているものを含めて合計 50 個提示する。それを参考にユーザーが変更後のコードを入力することでコード進行が更新される。

3　評価、考察

3.1　データベース

既存の曲の伴奏を学習内容とする。コードはメジャー、マイナーをはじめとしていくつも種類があり、さらにその種類×根音 12 音分のコードが存在する。種類が多いほどコードのパターンの選択肢が増え、より原曲に使用されているコードを再現することが可能になる。しかしその分データベースの収集、処理速度、結果の精度に影響が出るため、使用するコードの数を軽減させる必要がある。データベースに含まれる各コードが使用されている回数を基に頻繁に利用されているものを絞り、全ての音階についてのメジャー、マイナー、メジャーセブンス、ドミナントセブンス、マイナーセブンス、サスフォー、ディミニッシュ、シックス、マイナーシックスの 9 種類を用いていく事にする。尚、既存の曲からデータベースを作成する際に 6 種類以外が出てきた場合は、構成音が近いコードに置き換えて考える（例：Adim7 は Adim、Dm9 は Dm）。また、分数コードが用いられている場合、分母部分のベースとなる音は含めずに分子部分をコードとして扱う。

今回データベースは 2 種類作成する。1 つはメロディとコードの関係を表したもの（データベース 1）、もう 1 つはコードの遷移確率（データベース 2）である。データベース 1 については、あるコードが演奏されている間のメロディに含まれている音の分布をまとめたものである。コードはメロディの音に合わせて決められているため、ここからあるメロディがあったときに演奏可能となるコードを選択していく。データベースとして利用する曲目に対して、コードが演奏されている区間に含まれるメロディの音階 12 個の回数を求める。データベースの構築では、メロディとその最小音価、メロディに対するコード名、音価を入力して作成する。ビンの数はメロディの音階 12 個（C、C♯、D、D♯、E、F、F♯、G、G♯、A、A♯）として、図 4 のように利用するコード全てについてメロディの構成音の割合を求めていく。

データベース 2 では、使用されている全てのコードについて、あるコードからあるコードに遷移する確率を求めていく。この

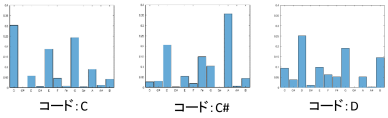

図 4　各コードのデータベースの分布 (縦軸：そのコードに含まれるメロディの音の割合、横軸：メロディの音階)

確率は、 データベース 1 を基に絞った選択肢の中から、実際の曲の中で使用確率の高いコードの遷移先を選ぶために用いる。尚、全ての曲について曲の始めの小節と終わりの小節にはそれぞれダミーのコード Y、Z を置いておく。この形で学習する事により、曲の初めと終わりにきやすいコードを推測することができる。入力されたメロディを処理する際はシステム上で前後に開始、終了用の小節をそれぞれ 1 小節ずつ追加し、その間に挟まれた部分が 1 曲分のコード進行を表すことになる。

3.2　HMM の学習結果

まず、学習後の HMM を確認する。和声法によると、コードはトニック (T)、サブドミナント (S)、ドミナント (D) の 3 種類に分類する事が出来る。[4] トニックはどの和音へも進行が可能であり、調性感を生み出し、安定感を感じる。曲の中での初めの小節、終わりの小節にはトニックが当てられる事がほとんどある。サブドミナントはトニック、ドミナントの両方へ進もうとする傾向にあり、トニックとの距離感とドミナントへの期待感を感じる。ドミナントはトニックへ進む傾向にあり、強い終始感を感じたり、転調による新しい調性感を感じる、といったように、各種類ごとに役割がある。基本的なコード進行はこの 3 つを組み合わせて出来ているが、中でも音楽的に多用される進行は次のようなものになる。

1. カデンツ第 1 型 T-D-T
2. カデンツ第 2 型 T-S-T
3. カデンツ第 3 型 T-D-S-T

データベース 2 でのコード遷移を学習した結果をみると、開始状態から移動する確率が高い状態、終了状態へ移動しやすい状態が一致していた。この状態は曲の初めと終わりに用いられることが多いトニックの役割をもっていると推測される。また、遷移確率の初期値を設定する際、状態の自己ループがある場合とない場合の比較を行った。自己ループがある場合、学習後の遷移確率を観察するとどの状態についても同じ状態に帰ってくる確率が非常に高くなり、実際の出力結果もずっと同じ状態からコードが出力されていた。トニック、サブドミナント、ドミナントはフレーズをまたぐ場合を除いて同じ分類が繰り返されることは少ないが、コードを見てみると同じコードが何小節も続くような箇所が多く見られた。自己ループなしの場合、ある状態からは遷移する確率が高い他の状態へ遷移しており、トニック、サブドミナント、ドミナントの分類としても無意味に連続する箇所は見られなかった。よって、遷移確率の初期値は自己ループがないように設定を行うのが適切だといえる。

3.3　コード進行の出力結果

実際にメロディを読み込んでコード進行を作成したものを観察する。コードを付けたいメロディの情報を楽譜から読み込み、プログラムで利用できるように処理をしていく。必要な情報はメロディの音階と音価、その楽譜内での最小音価である。音階については、コードの決定にオクターブ情報は必要なく、音階の情報のみを取り出していく。曲全体を読み込んだら、一定区間毎にメロディを取り出して音の分布をヒストグラムにしていく。実際の曲において、コードが何拍毎に変わるかは決まっていない。しかし、どのタイミングで音を切り替えるかをシステムが判断する事は難しいため、コードを切り替えたい拍数をユーザが指定し、その拍数毎にメロディを解析してコードを付与していく事にする。今回の評価実験においては 1 小節毎にコードを付与するよう設定した。

作成されたコード進行について、既存の曲の伴奏の再現度の評価、完成した伴奏の主観評価の 2 つを行う。再現度については楽譜がある曲でシステムを利用し、 元の伴奏と比較をしてコードがどれくらい同じように付与されているかを求めて従来システムと比較する。 先行研究である MySong、既存のソフトである Band in a Box でも同じメロディで伴奏作成を行い、今回の提案システムとの比較を行った。コードの中には代理コードというものがある。代理コードとは、C と Am、F と D m のように共通した構成音を 2 つ以上含み似たような響きを持っている為、代わりに用いる事の出来るコードである。このように、原曲と完全に一致していなかったとしても適切であるコードとなる可能性があるため、構成音当たりで計算した再現率も求める。例として、原曲が C、出力が Am だった場合、C の構成音は「C E G」、Am の構成音は「A　C　E」であるため、3 音中 2 音を含んでいるとして一致率は $\frac{2}{3}$ として計算する。同様に、原曲と出力のコードがそれぞれ C(構成音：C　E　G) と G(構成音：G C D) の場合は $\frac{1}{3}$、G(構成音：G C D) と G7(構成音：G C D F) の場合は $\frac{3}{4}$、といったように各コードの一致率を求めていき、1 曲分の一致率の合計をコードが出てきた回数で割った値が 1 曲分の再現率とし、その結果を表 1 に示す。尚、今回ヴォイシング [5] は考慮しないものとする。

表 1　既存のコード進行の再現度 (コード構成音当たりの場合)

曲名	プログラム	コードの一致率
It's a small world	BIAB	62.7%
	MySong	43.1%
	提案システム	92.2%
Autumn Leaves	BIAB	44.8%
	MySong	35.3%
	提案システム	59.3%
The Glory of Love	BIAB	58.9%
	MySong	52.2%
	提案システム	82.5%
It could happen to you	BIAB	55.0%
	MySong	42.5%
	提案システム	53.1%
The Peanut Vendor	BIAB	50.3%
	MySong	36.1%
	提案システム	74.0%
平均 (5 曲分)	BIAB	54.3%
	MySong	41.8%
	提案システム	72.2%

原曲との一致率が低かった曲を確認すると、臨時記号を含むメロディに対して合うコードが出てくる確率が低い。コード遷移のデータベースで用いている曲は全て C のキーに移調している為、出力確率では高い値を得ていても遷移確率の部分ではじかれてしまい、出力に現れなかったと考えられる。また、構成音が似ているコードの付与も特徴として挙げられる。構成音単位で再現率を求めている為、あるコードが出る度に構成音が何音か違うコードが出力されているような曲については再現率が下がってしまっている。しかし、原曲のコードとの一致率が高いものであれば聞いた時の違和感はあまり感じないコード進行を作成する事が出来る。他の翌朝として、コードを付与するタイミングの違いが挙げられる。提案システムではコードの切り替えのタイミングは固定であるが、実際の曲では 1 小節に 2 回コードがあてられている箇所もあり、その部分でも再現率の低下につながった。提案システムでも固定であれば何拍毎にコードを切り替えるか指定をする事は出来るが、2 拍や 1 拍毎にコードをつけるとすると、その区間に含まれるメロディの音数が少なくなってしまい、メロディに対して適切なコードの判別が難しくなり精度が下がってしまうという問題が残る。

また、コードの分類が行えているかを確認するためにコードの分類ごとでの比較も行った。トニック、サブドミナント、ドミナントに分類できるコードについてそれぞれのシステムで当てられたコード進行について分類を観察したところ、図 5 のようになった。

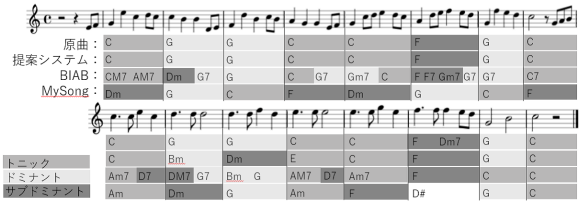

図 5　コード分類単位での比較

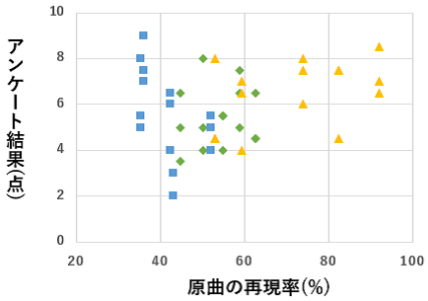

図 6　アンケート結果と再現率の関係

不一致のコードに対して分類も違うコードの割合を求めると、提案システムが 25.0% 、Band In A Box が 38.1% 、MySong が 83.3% 提案であった。提案システムではコードは誤っていても同じ分類の中から出力されていることが多く、トニック、サブドミナント、ドミナントにある程度沿うように遷移確率が学習されているといえる。また、それぞれのコードはどの状態から出力されたものであるかを確認すると、トニックとドミナントが同じ状態として分類されている傾向にあった。今回実装を行った際にはメロディに対するコードの出力尤度を 0.8 倍、各状態での出力確率を 0.2 倍としているが、これをそれぞれ 0.7 倍、0.3 倍としてみると、トニックとドミナントも概ね別の状態として分類できている事が確認できた。よって、重みづけによって結果が変わりやすく不安定な状態であるが、HMM によってコードの役割の分類を学習する事が出来ているといえる。続いて、主観評価としてアンケートを実施した。従来システムと提案システムでコードを作成した 5 曲について被験者に聞いてもらい、「メロディの音に対してコードが適切だと思うか」「コード進行は適切だと思うか」という質問にそれぞれ 10 点満点で答えてもらった (表 2)。アンケート結果を見ると、メロディに対してのコードの適合度が平均して Band In A Box が 5.4 点、MySong が 5.2 点、提案システムが 6.7 点となった。また、コード進行の適切さについてに平均はそれぞれ 5.5 点、5.5 点、6.6 点であり、曲による差はあるもののどちらの項目についても提案システムが高い評価を得た。

表 2　コード進行の完成度についてのアンケート結果 (1:メロディの音への適合度、2:コード進行の適切さ/被験者 3 名による点数の平均)

曲名	プログラム	1	2
It's a small world	BIAB	5.0 点	5.3 点
	MySong	2.0 点	2.7 点
	提案シス	7.7 点	7.0 点
Autumn Leaves	BIAB	5.0 点	5.0 点
	MySong	5.0 点	6.0 点
	提案シス	5.7 点	6.0 点
The Glory of Love	BIAB	6.3 点	6.3 点
	MySong	4.7 点	5.0 点
	提案シス	7.0 点	7.3 点
It could happen to you	BIAB	5.3 点	5.7 点
	MySong	5.3 点	4.7 点
	提案シス	5.7 点	5.7 点
The Peanut Vendor	BIAB	4.0 点	6.0 点
	MySong	7.7 点	8.0 点
	提案シス	7.3 点	7.0 点
平均 (5 曲分)	BIAB	5.1 点	5.7 点
	MySong	4.9 点	5.2 点
	提案シス	6.7 点	6.6 点

また、アンケート結果について各曲と構成音単位での再現率を比較した。各曲について、二つの質問の点数の平均とコードの再現度の散布図を図 6 に示す。右肩上がりの傾向にあり、コードの再現率が高いほど被験者の評価も高い事が読み取れる。よって、構成音単位での再現率を完成度として評価することは有効であるといえる。

以上の結果を踏まえて各コードについて確認すると、原曲のコードの出力確率は比較的高い値を得る事が出来ていた。これにより、不一致だった部分もメロディに対するコードは概ね正しく求める事が出来ていると言える。よって、コード遷移の尤度の部分が不十分であると考えられる。構成音が似ているコードを出力している場合以外の不一致だった箇所をみると、臨時記号を含むメロディに対して合うコードが出てくる確率が低い。コード遷移のデータベースで用いている曲は全て C のキーに移調している為、出力確率では高い値を得ていても遷移確率の部分ではじかれてしまい、出力に現れなかったと考えられる。

4　おわりに

本稿では、あるメロディが与えられた時に伴奏として適切なコード進行を出力するシステムを提案した。メロディの構成音に対して付与可能なコードと、そのコードから次のコードへの遷移の仕方を既存の曲に基づいて選択していくことで、利用者に知識や伴奏を付けた経験が無くてもコード進行を提示する事が可能になった。今後の課題としては、条件による出力結果の差を無くす事が挙げられる。コード遷移のデータベースを C に限定したことにより、メロディが C のキーのスケール内で構成されていれば完成度の高いコード進行を作成出来るようになった。しかしその反面で、C のキーから外れたメロディが出てきた場合、C のキーではあまり出てこないコードの出力確率が低くなってしまい、イレギュラーな場合への対応が出来なくなってしまった。コード遷移のデータベースを増やすことで改善が期待できるが、ただ増やすだけではなく、含まれているコードの種類も考慮する必要がある。また、現時点ではコードの状態出力確率を乱数で定めており、トポロジーを変えた際に適当な初期値を得るまで何度か学習を行う必要がある。そのため、どんなトポロジーであっても適切な結果を得られるような初期値を設定できるようにする必要がある。

参考文献

[1] Ian Simon, Dan Morris, Sumit Basu, “MySong: Automatic Accompaniment Generation for Vocal Melodies”, CHI ’08 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems , 2008, pp.725-734.

[2] Ian Simon, Dan Morris, Sumit Basu, ”Exposing Parameters of a Trained Dynamic Model for Interactive Music Creation”

[3] 山本寛樹, 村上仁一, 嵯峨山茂樹, ”隠れマルコフモデルによる言語モデル自動獲得の検討”, 全国大会講演論文集, 1992, pp.227-228

[4] 金森務, 平井宏, 堤喜代司, 弓場芳治, 新美康永, ”コード及びメロディー・パートからの感性情報の抽出”, 情報処理学会研究報告ヒューマンコンピュータインタラクション, 1992, pp.75-79

[5] 北原鉄朗, 勝占真規子, 片寄晴弘, 長田典子, ”ベイジアンネットワークを用いた自動コードヴォイシングシステム”, 情報処理学会論文誌, 2009, pp.1067-1078