フレーム同期型フルサーチアルゴリズムを用いた連続音声認識と自由発話への応用

村上仁一

NTT 情報通信研究所
(ATR 音声翻訳通信研究所)

神奈川県横須賀市武 1-2356

あらまし

人間同士のコミュニケーションでは、「あのー」「えーと」などに代表される間投詞や、言い淀みや言い誤りおよび言い直しなどが頻繁に出現する。このような音声でも認識できる、いわゆる自由発話の音声認識が、今後の重要な研究課題になると思われる。しかし、間投詞や言い直しは、文の全ての場所に出現する可能性がある。また、自由発話様式において認識精度の高い音響モデルを作成することは困難であると考えられる。そこで本論文では perplexity の低い言語モデルに着目した。そして単語の trigram モデルを基本の言語モデルとして、間投詞や言い直しを garbage model や音素モデルでスキップすることで自由発話の認識を試みた。

この論文では、初めにフレーム同期型のフルサーチアルゴリズム (全探索) を用いた連続音声認識システムについて述べる。次に計算量およびメモリ量を削減するための改良点について述べる。次に自由発話の認識のためのアルゴリズムの改良点を述べる。最後に、このアルゴリズムを用いた自由発話の認識実験結果について述べる。

キーワード • 自由発話認識 • 全探索 • ビームサーチ • 時間同期 • ガーベージモデル • 単語 trigram model

Frame Synchronous Full Search Algorithm and Applied for Spontaneous Speech Recognition

村上仁一

NTT Information and Communication System Laboratories
(ATR Interpreting Telecommunications Research Laboratories)

1-2356 Take Yokosuka-shi Kanagawa 238-03 Japan

abstract

This paper describes an spontanous speech recognition algorithm based on word trigram models.

This recognition algorithm greatly reduces the memory requirements and computational costs by employing beam search. With these methods, we can perform execution in a 15M byte space for about a 500-word vocabulary. Next, focusing on spontaneous speech recognition, we look at a skip phone procedure to handle the many filled pauses and false starts in spontaneous speech. Even though the proposed method employs a simple procedure, we obtain a 47.7% sentence recognition rate for spontaneous speech. Including the semantically correct sentences, the sentence recognition rate is about 75%.

key words • Spontaneous Speech Recognition • Full Search • Beam Search • Frame Synchronous • Garbage Model • Word trigram Model

1 まえがき

従来の多くの音声認識システムは、丁寧に発声された音声を入力対象にしている。しかし、人間同士のコミュニケーションでは、「あのー」「えーと」などに代表される間投詞や、言い淀みや言い誤りおよび言い直しなどが頻繁に出現する。このような音声でも認識できる、いわゆる自由発話の音声認識が、今後の重要な研究課題になると思われる。しかし、このような発話様式において認識精度の高い音響モデルを作成することは困難であると考えられる。そこで perplexity の低い言語モデルに着目した。

現在、多くのシステムでは簡潔さ・有効などの点から単語の bigram モデルが音声認識のための言語モデルとして使用されている [1]。しかし一般的に単語の trigram モデルは bigram と比較すると低い perplexity を持つことが知られている [2]。そこで基本的な言語モデルとして、単語の trigram を利用した。また、自由発話では対話文の 50%に「あのー」、「えーと」などの間投詞を含む。また言い直しは約 10%に出現する [3]。これらの間投詞や言い直しは、文の全ての場所に出現する可能性がある。そこで garbage model や音素モデルによって、これらの言語現象をスキップすることで間投詞や、言い淀みや言い誤りおよび言い直しなどを含む発話の認識が可能になる。

この論文では、初めに連続音声認識アルゴリズムとしてフレーム同期型のフルサーチアルゴリズム (全探索) について述べる。連続音声認識のアルゴリズムには One-Pass DP や level Buliding や 2 段 DP 法などが知られている。しかし、本論文では各フレームごとに単語の全ての組み合わせを探索するフレーム同期型のフルサーチ (全探索) を採用した。フルサーチアルゴリズムはグリッドを trellis で計算できることや、N-best リストが計算できるなどの長所を持つ。しかし大量の計算時間とメモリー量が必要になるという短所も持つ。そこで、計算時間とメモリー量を削減するためにアルゴリズムを改良した。これらは、ビームサーチの採用とその改良に依存している。次に自由発話の認識のためにアルゴリズムを改良した。基本的な言語モデルとして、単語の trigram を利用した。また間投詞や言い直しは garbage model や音素モデルによってスキップすることで自由発話の認識ができるようにした。最後に、このアルゴリズムを用いた自由発話の認識実験結果について報告する。

2 フレーム同期型フルサーチアルゴリズム

フルサーチアルゴリズム (全探索) は連続音声認識のための基本的なアルゴリズムである。このアルゴリズムは全ての単語の組み合わせの尤度を計算するため、大量の計算量やメモリ量が必要になる。

しかし、以下に示すような長所を持っている。

1. N 位までの累積尤度の単語列 (N-best リスト) を出力することができる。
2. グリッドの選択において、最尤なものを選択する方法 (Viterbi) と尤度を足す方法 (Trellis) の両者が選択できる。
3. グリッドを Trellis で計算をした場合、状態を陽に考慮する必要がないため、継続時間のコントロールは基本的に必要としない。
4. 音声データの最終フレームにおいて、尤度の最も高いグリッドを選択して単語系列を認識するため、基本的に traceback は必要としない。
5. フレーム同期で計算を実行させたとき、任意の時間において最尤の単語を認識できるため、単語スポッタとしての動作が可能である。
6. 各単語の HMM の最終状態を示すグリッドと後続する単語の最初の状態を示すグリッドの尤度計算において、任意の言語モデルの制約を加えることにより、left-right 型の言語モデルと音響モデルを容易に結合することができる。

図 1 に、フルサーチアルゴリズムをフレーム同期で計算したときの簡略図を示す。この図では、認識語彙数を w_a と w_b の 2Word で、単語を認識単位とし、4-state 3-loop の HMM、状態は 0 から 2 とする。縦軸は HMM の状態で、横軸は時間で、奥行きは語彙を示している。

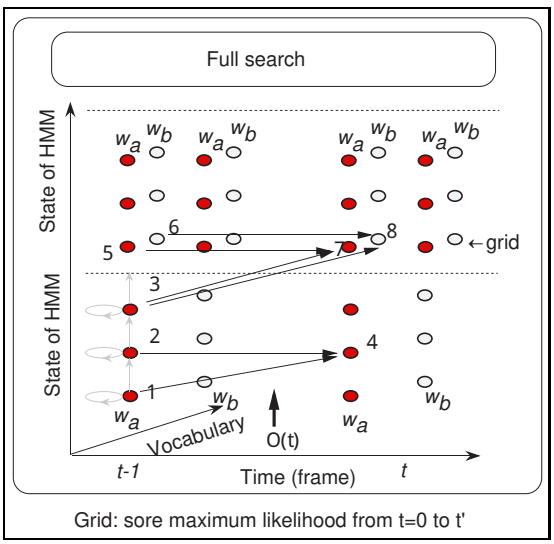

Figure 1: フルサーチのアルゴリズム

図中の○1は時間 0 から時間 $t-1$ までの単語 w_a の状態 0、○2は時間 0 から時間 $t-1$ までの単語 w_a の状態 1、○3は時間 0 から時間 $t-1$ までの単語 w_a の状態 2、○4は時間 0 から時間 t までの単語 w_a の

状態 2、○5は時間 0 から時間 $t-1$ までの連続 2 単語 w_a, w_a の状態 0、○6は時間 0 から時間 $t-1$ までの連続 2 単語 w_a, w_b の状態 0、○7は時間 0 から時間 t までの連続 2 単語 w_a, w_a の状態 0、○8は時間 0 から時間 t までの連続 2 単語 w_a, w_b の状態 0、を示すグリッドで、累積尤度が記録されている。

フレーム同期型フルサーチアルゴリズムにおいてグリッドを trellis で計算する場合、単語の最初の状態 0 を示すグリッド以外は、前時刻の同一状態を示すグリッドの尤度と前時刻の 1 つ前の状態のグリッドの尤度を加算して現時刻のグリッドの尤度を計算する。例えば、○4は○1の遷移と○2から遷移の累積尤度の総和とする。しかし、単語の最初の状態 0 を示すグリッドは、前時刻の同一のグリッドの累積尤度とレベルが 1 つ下の単語の最終状態を示すグリッドの累積尤度の総和とする。例えば、○7は○3の遷移と○5の遷移の累積尤度の総和とする。これを全グリッドに対して計算を行なう。

また、単語の遷移において言語モデルの確率値を掛けることにより音響モデルと言語モデルが結合できる。例えば、○7は○3の遷移と○5の遷移の累積尤度に単語 bigram$P(w_b|w_a)^{\alpha}$, (α: language weight) を掛けることによって、単語の bigram と単語の HMM が容易に結合できる。

2.1 計算量およびメモリ量を削減したアルゴリズム

実際の認識実験においてフルサーチアルゴリズム (全探索) を使用した場合、大量のメモリと計算量が必要になる。そこでメモリ量と計算量を削減するために、アルゴリズムを次に述べるように改良する。

2.2 ビームサーチ

各フレームごとの尤度計算において、累積尤度の低い単語列は正解の単語列になる可能性が低いため、以後の探索から除外できる可能性が高い。そこで、フレームごとに最も高い累積尤度から正解の存在をおおよそ保証できる、ある個数 (ビーム幅 b) のみ計算を続けることにより、計算量およびメモリ量が削減できる [4]。具体的には、すべての $w_n, .., w_0, i$ に対して最も高い累積尤度から、ある個数 (ビーム幅 b) のみを計算する。したがって $G_t(w_n, ..., w_0, i)$ を記憶するメモリ量は、フルサーチでは O(認識語彙数 $^n\times$ 単語の状態数) が必要であるのに対し、ビームサーチでは O(ビーム幅 b) しか必要としないため大幅に削減できる。また、計算量もビーム幅の計算方法によって異なるが、同様な比率で削減できる。

なお、フルサーチをフレーム同期で計算し、さらにビームサーチと組み合わせたアルゴリズムは、フレーム同期型の stack decorder[5] とアルゴリズムは同一であると考えている。

2.3 ビームの絞り方

ビームの絞り方には、次の 2 つの方法がある。

1. 尤度の閾値

   尤度の閾値でビームを絞る方法は、計算量が少なくてすむためよく利用されている [1]。しかし、認識を行なう前に予め閾値を決めておかなければならないため、動作が不安定になることがある。

2. ビーム幅

   一定のビーム幅でビームを絞る方法は、フレームごとにソーティングが必要になる。そのため、計算量が増大する。

本論文では、後者のアルゴリズムを選択している。だたし、計算順序として、始めにフレームごとに最も高い最大累積尤度 (G_{max}) からビーム幅 b の最大累積尤度 (G_b) を計算し、次にこの尤度 (G_b) でビームを絞り込んでいる。この方法はフルソートと比較すると計算量が大幅に削減できる [6]。

2.4 近接したフレームにおける文仮説候補の類似性の利用

フレーム同期型の連続音声認識アルゴリズムでは、時刻 t において言語モデルによって生成された文仮説候補は、次のフレーム (時刻 $t+1$) においても文仮説候補になる可能性が高い。この特徴を利用することで、言語モデルに必要な計算時間が大幅に削減できる。なお、この方法は文献 [7] においても紹介されている。

2.5 log 計算

グリットの trellis 計算では、対数尤度の足し算が必要になる。このアルゴリズムとして、文献 [1] において線形補間の方法が報告されている。ここでは以下に述べる方法を採用している。

$A = \log(a)$, $B = \log(b)$ として

$C = \log(a+b)$

$= A + \log(1 + \exp(B - A))$

3 自由発話の音声認識アルゴリズム

人間同士のコミュニケーションでは、「あのー」「えーと」などに代表される間投詞や、言い淀みや言い誤りおよび言い直しなどが頻繁に出現する。しかし、このような発話様式において認識精度の高い音響モデルを作成することは困難であると考えられる。そこで perplexity の低い言語モデルに着目し、単語 trigram を基本の言語モデルとした。

また、自由発話に特有な言語現象として間投詞や言い淀み・言い直しがある。これらの間投詞や

言い直しは、文の全ての場所に出現する可能性がある。これらの対応方法として連続 Viterbi アルゴリズムによるキーワードスポッテングなどの多くの方法が提案されている。これらのアルゴリズムの中の 1 つに garbage モデルを使用する方法がある。garbage モデルは、キーワードスポッテングにおいて使用されていたモデルで、キーワード以外の音素を数個の HMM でモデル化する [8][9]。これを Viterbi サーチに組み込み、間投詞や言い直しなどの不要語を garbage モデルで対応する方法を井上らは提案している [10]。

この方法は、間投詞や言い直しを音響モデルで解決した方法と言える。しかし、言語モデルにおいて、間投詞や言い直しを音素の系列とみなし、この音素をスキップすることで同様なことが実現できる。

本論文では単語の trigram モデルを基本の言語モデルとして、この 2 つの方法で自由発話の認識を試みた。

3.1 garbage モデル (音響モデルによる対策)

garbage モデルは、間投詞や言い直しを 1 個ないし数個の garbage HMM で処理する方法である。garbage HMM には予め全ての音素を学習しておく。そして認識アルゴリズムにおいて、このモデルは 1 つの単語として扱う。また単語の trigram の連鎖確率値は garbage モデルをスキップをして計算をする。この結果、音声データ中の間投詞や言い直しは garbage モデルで認識しながら、言語モデルではこれらの言語現象をスキップすることで、間投詞や言い直しを含む自由発話の音声でも認識が可能になる [10]。

3.2 音素スキップ (言語モデルによる対策)

間投詞や言い直しは、文の全ての場所に出現する可能性がある。そこで、音響モデルでは間投詞や言い直しを音素系列として認識しながら、言語モデルでは音素系列をスキップすることにより、間投詞や言い直しを含む自由発話の音声でも認識が可能になる。ただし、このようなアルゴリズムでは、音声データ全てが音素系列と認識される可能性があるため、本論文ではペナルティとして音素の trigram を使用する。

例えば「“東京都” “港区” “新橋” “あのう (anou)” “1 丁目” 」と発声されたとする。そして「あのう」は間投詞とする。このときの 言語モデルの連鎖確率値は $P($ “新橋” | “東京都”, “港区”$) \times P(/a/|/sh/,/i/) \times P(/n/|/i/,/a/) \times P(/o/|/a/,/n/) \times P(/u/|/n/,/o/) \times P($“1 丁目” | “港区” “新橋”) と計算する。ここで、$P(/a/|/sh/,/i/)$ はペナルティ、 $P($“1 丁目” | “港区” “新橋”) は” あのう” を音素系列と見てスキップしたことを示している。

この方法は、前節で述べた garbage モデルを言語モデルで実現する方法であると言える。また、既に提案されている未知語検出のアルゴリズムと基本的には同一の思想である [11],[12],[7],[13],[14],[16] 。ただし、これらの論文では未知語検出を目的にしている。また、使用している言語モデルも異なる。

4 自由発話認識

4.1 自由発話認識の実験条件

上記で提案したアルゴリズムの有効性を調べるために、自由発話の文認識実験をおこなった。実験には、音響モデルには不特定話者の HMM、言語モデルには単語の trigram を使用した。実験条件を表 1 に示す。garbage モデルは、4 状態 3 ループの 10 混合のモデルで、男性話者 12 名の音素バランス 216 単語から作成した。単語の trigram の連鎖確率値は、 ATR の対話データベース [17] のなかから国際会議の予約に関するデータ約 1 万 2 千文章、約 17 万単語にテストデータの中の朗読発話のテキスト (次節参照) を加えて計算した。ただしテキストデータ中の「あのー」、「えーと」などの間投詞は削除している。また、音素の trigram の連鎖確率値は「あのー」、「えーと」などの間投詞を含めて同一の ATR の国際会議の予約のデータベースから作成した。

Table 1: 自由発話認識の実験条件

音素モデル	4-state 3-loop
(garbage model を含む)	10mixture left-right model
HMM の学習データ	男性話者 12 名の 736 単語発声
garbage モデル	男性話者 12 名の
の学習データ	音韻バランス 216 単語
認識単語数	435
ビーム幅	16,384
単語 trigram の値の	約 1 万 2 千文章 約 170000 単語
推定に使用した	朗読発話のテキストを含む
テキストデータ量	(間投詞は削除)
音素 trigram の値の	約 1 万 2 千文章 約 170000 単語
推定に使用した	朗読発話のテキストを含む
テキストデータ量	(間投詞を含む)
言語尤度と音響尤度の	16
結合値 α (考察参照)	

4.2 自由発話の音声データ

音声データは以下に示すような方法で収録した。ただし話者は一般人である。

1. 朗読発話

   テキストを読みあげた音声データ。間投詞や言い淀み・言い直しは無い。なお、実験に使用した単語の trigram の連鎖確率値は、このテキストを加えて計算した。したがって、こ

のデータの認識実験は、言語モデルに対して text-closed データの実験になる。

2. 疑似自由発話

   間投詞を含むテキストを読みあげた音声データ。間投詞を除いて、「1 朗読発話」と発話内容は同一。言い淀み・言い直しは無い。

3. 自由発話

   話者はテキストを覚えて、その意図を理解してから自由に発話した音声データ。発話内容は「1 朗読発話」と異なる。間投詞や言い直しや未知語を含む。このデータは、言語モデルに対して text-open のデータと言えるが、テキストを覚えて発話したデータであるため、発話内容は text-closed データに近い。

4.3 単語の trigram の連鎖確率値の平滑化

単語 trigram は語彙数の 3 乗のパラメータの数をもつ。したがって全ての trigram の値を直接推定できるだけの大量のテキストデータを収集することは困難である。そのため通常 trigram の連鎖確率値を平滑化して使用する。ここでは deleted-interpolation[15] を使用した。そして、単語の trigram の値を平滑化した場合としなかった場合の両方で認識実験を行った。

4.4 自由発話の認識実験結果

表 2 に単語の trigram の連鎖確率値を平滑化しない場合の認識実験結果を示す。また、表 3 に単語の trigram の連鎖確率値を deleted-interpolation で平滑化して認識実験を行なった結果を示す。図中、base-line は単純に単語の trigram モデルを用いた実験結果、garbage は単語の trigram モデルと garbage モデルを組み合わせた実験結果、skip-phone は単語の trigram モデルと音素 trigram モデルを組み合わせた実験結果を示している。また、Word Correct と Word Accuracy は第 1 位認識候補に対して HTK[18] を使用して計算した。なお、平滑化の値は、trigram,bigram,unigram, フロアリングに対して各々$\lambda_3 = 0.35, \lambda_2 = 0.48, \lambda_1 = 0.11, \lambda_0 = 0.06$ となった。

Table 2: 文認識実験結果（平滑化無し）
文認識率 (%)

	累積文認識率	base-line	garbage	skip-phone
朗読発話	1	89.7%	83.2%	88.5%
	~ 2	97.3%	90.5%	96.2%
	~ 8	100.0%	97.3%	99.2%
	Word-Correct	97.5%	93.4%	96.4%
	Word-Accuracy	96.9%	93.2%	96.0%
疑似自由発話	1	41.6%	64.5%	73.3%
	~ 2	43.1%	70.2%	79.0%
	~ 8	44.3%	78.2%	82.8%
	Word-Correct	70.6%	81.5%	89.5%
	Word-Accuracy	34.2%	76.6%	82.3%
自由発話	1	10.7%	37.8%	47.7%
	~ 2	15.3%	46.9%	57.2%
	~ 8	19.5%	56.1%	66.8%
	Word-Correct	44.7%	65.7%	80.9%
	Word-Accuracy	9.2%	58.9%	73.3%

不特定話者認識; 語彙数:435; ビーム幅:16,384; α:16
trigram の連鎖確率を直接使用

Table 3: 文認識実験結果（平滑化有り）
文認識率 (%)

	累積文認識率	base-line	garbage	skip-phone
朗読発話	1	47.3%	49.2%	46.9%
	~ 2	52.2%	53.8%	53.4%
	~ 8	61.5%	61.8%	64.1%
	Word-Correct	77.4%	70.1%	77.1%
	Word-Accuracy	71.9%	68.0%	72.2%
疑似自由発話	1	29.0%	36.3%	28.6%
	~ 2	30.1%	37.8%	30.9%
	~ 8	33.2%	42.0%	36.3%
	Word-Correct	63.1%	59.6%	63.2%
	Word-Accuracy	28.8%	44.3%	29.9%
自由発話	1	10.3%	16.4%	10.7%
	~ 2	14.1%	18.3%	13.0%
	~ 8	17.5%	22.1%	16.8%
	Word-Correct	51.0%	41.5%	46.7%
	Word-Accuracy	27.9%	26.5%	19.2%

不特定話者認識; 語彙数:435; ビーム幅:16,384; α:16
trigram の連鎖確率を deleted-interpolation で平滑化して使用

これらの実験から以下のことがわかる。

1. 自由発話認識においては、音素スキップを使用した場合も、garbage モデルを使用した場合も、単純な単語の trigram モデルより認識性能は向上する。

2. garbage モデルを使用したときと音素スキップを利用したときの認識率を比較すると、音素スキップを利用したときの方が高い認識性能を得ている。

3. 単語の trigram の連鎖確率値を平滑化した場合と平滑化しない場合の認識率を比較すると、自由発話の認識では平滑化をしないほうが高い認識性能を得ている。

4. 単語の trigram の連鎖確率値の平滑化を行なわないで、音素スキップを利用することで、自由発話では 47.7%の文認識率が得られた。また朗読発話でも、この処理を加えることの認識性能の低下は少なかった (89.7% → 88.5%)。

自由発話において音素スキップを使用したときの誤認識の例を、表 4 に載せる。表中の括弧内は、実際の発話内容である。またアンダーラインは誤認識の箇所を示している。

Table 4: 実験において誤りが出力された文 (自由発話認識)

(ビーム幅:16,384; α:16)

正解文 (発声内容) → 1 位出力
会議の宿泊施設についてお尋ねしたいのですが
(会議の宿泊施設についてお尋ねしたいんですけれど)
→ 会議の宿泊施設についてお尋ねしたい んですよ
私共でご紹介できるホテルは京都ホテルと...
(えーと、私共でご紹介できるホテルは京都ホテルと...)
→ 登録を ご紹介できるホテルは京都ホテルと...
ではお名前と住所お願いします
(ではお名前と住所お願いします)
→ ではお名前と ご 住所をお願いします
会議の参加料について教えていただきたいのですが
(えー、会議の参加料について教えていただきたいのですけれども)
→ 会議の参加料について教えて いただけますか
失礼します
(う、失礼します)
→ そうします
京都プリンスホテルに 8 月 4 日から 8 日まで一人部屋を...
(えーっと、京都プリンスホテルに 8 月 4 日から 8 日まで...
→ 国際会議が 8 月に行われているんでしょうか

自由発話では発話内容が朗読発話と異なる。しかし、文認識率の計算においては、朗読発話の単語と一致した場合に正解とした。そのため、発話内容と認識結果が合っていても誤認識とした。したがって意味的に正しいとみなされる文を正解に含めた場合、1 位文理解率で約 75%、8 位までの累積文理解率は 90% と理解率は大きく向上する。

これらの実験の結果、音素スキップの方法は、自由発話の認識において有効であることが示された。

5 考察

5.1 フルサーチと One-pass サーチの比較

連続音声認識アルゴリズムとしては One-pass サーチがよく知られている。しかし、表 1 の実験条件では、フルサーチのほうが僅かに高い認識性能 (文認識率において約 2%) が得られた。しかし、この差は、ビーム幅や語彙数によって変化すると考えている。表 5 に、このフルサーチと One-pass サーチの違いについてまとめた。

なお、グリッドを中心に考えると、フルサーチにおいてグリットの選択方法を Viterbi にし、生成された単語仮説を、単語ごとに選択をすると (つまり $localbeam = 1$)、One-pass サーチになる。また、前後の音素環境も考慮しながらグリッドを選択することにより triphone のような環境依存音素モデルが扱える。また、グリッドを、言語モデルからみて過去の単語系列や構文解析木が一致する場合のみ選択することで、ネットワーク文法や文脈依存文法などの left-right 型の言語モデルと音響モデルを組み込むことができる。

従ってグリッドを中心に考えることにより、フルサーチと One-pass サーチ、そして環境依存音素モデル、さらに left-right 型の言語モデルが統一できる。これをオブジェクトグリットと呼んでいる。

5.2 音響尤度と言語の連鎖確率の結合値 α

言語情報として単語の trigram を用いた場合、求める解は文候補 $l(w_1, w_2, .., w_N)$ を以下のように定式化して、これを最大にする文 (単語系列) $w_1, w_2, ..., w_N$ を選び出すことである。

$$\sum \log(P_a(w_i)) + \alpha \times \sum \log(P(w_i|w_{i-2}, w_{i-1}))$$

ここで $P_a(w_i)$ は単語 w_i の音響尤度、$P(w_i|w_{i-2}, w_{i-1})$ は単語 w_{i-2} の次に単語 w_{i-1} が現れたときに w_i に遷移する確率、α は音響尤度と単語の尤度を結びつける結合定数である。

ここでは音響尤度と言語の連鎖確率の結合値 α を変化させたときの文認識率の変化を調べた。テストデータは朗読発話で語彙数は 1500 ビーム幅は 4096 である。他の実験条件は表 1 と同一である。この結果を図 2 に示す。この図において横軸は結合値 α で、この値が大きいほど言語尤度が音響尤度と比較して増加する。縦軸は文認識率である。

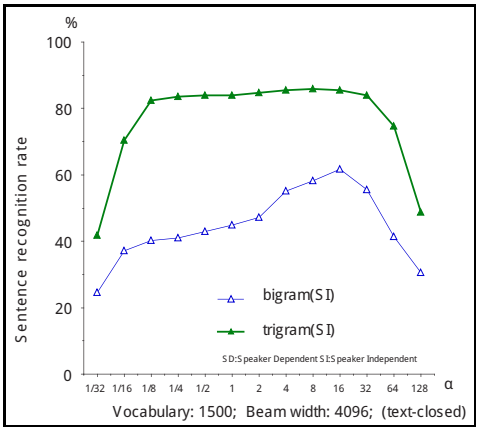

Figure 2: 音響尤度と言語の連鎖確率の結合値を変えたときの認識性能の変化　文認識率 (%)
Sentence recognition rate versus language weight

この実験から音響尤度と言語の連鎖確率の結合値 α が 16 のとき最も高い文認識率が得られた。

ただし、個人的には音響尤度と言語の連鎖確率の結合値は 1 であると考えている。その根拠は以下の通りである。

Table 5: フルサーチと One Pass サーチの比較

	フルサーチ	One-pass サーチ
計算コスト	大きい	小さい
メモリ	大きい	小さい
グリッドの選択方法	Viterbi & trellis	Viteribi
N-best list	可能	アルゴリズムを改良して可能
言語モデルとの適合性	left-right 型の全ての言語モデルが可能	left-right 型の全ての言語モデルが可能。ただし近似解になる。
ビームサーチとの適合性	良好	良好
音素モデルにおける duration control との適合性	良好 (ただし Trellis 計算では必要としない。)	良好
スポッタとしての動作	可能	プログラムを改良すれば可能

単語の HMM と単語の bigram を考えて、これらを組み合わせたモデルは ErgodicHMM となる。そして単語の bigram の値は 1 つの単語の HMM の最終状態の遷移確率を別の単語に接続されたときの値の分配率になる (図 3)。例えば Word_a の HMM の最終状態の状態遷移確率が 0.5 で、Word_a から Word_a の bigram が 0.4 Word_a から Word_b の bigram が 0.6 であったとする。このとき単語の HMM と単語の bigram を組み合わせた ErgodicHMM において Word_a から Word_a の状態遷移確率は $0.5 \times 0.4 = 0.2$ Word_a から Word_b の状態遷移確率は $0.5 \times 0.6 = 0.3$ が、もっとも妥当であると考えられる。

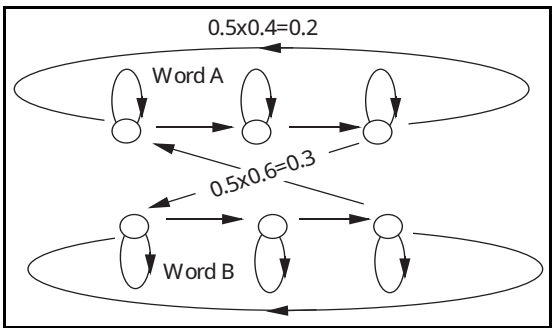

Figure 3: 音響尤度と言語の連鎖確率の結合値 $\alpha = 1$ の根拠

5.3 単語の trigram の連鎖確率値の平滑化に関して

今回の自由発話の実験では、単語の trigram の値を平滑化をしない方が、deleted-interpolation で平滑化をした場合より高い認識性能が得られた。この原因を deleteted-interpolation の計算方法にあると考えている。deleteted-interpolation は、有限個のテキストから単語の unigram と bigram と trigram の連鎖確率値を計算し、真の trigram の連鎖確率値をこれらの組合せで推定できると仮定して、この比を同じテキストデータから計算している [15]。つまり garbage model や音素スキップを想定していない。したがって garbage model や音素スキップなどを採用した認識システムでは、通常の deleted-interpolation で求めた値が妥当な値ではなくなると考えられる。したがって認識性能が低下したと考えられる。今後、garbage model や音素スキップなどを取り入れたときの deleted-interpolation の方法を検討する必要がある。

なお、自由発話では、文字化した文章と発話した音素列の差は朗読発話より大きくなる。例えば「会議にー (い)」と発声している音声を「会議に」と文字化している。したがって、自由発話の音声認識では、全ての音素を完全に認識することはできなくて、意味的に合っている文章を出力することしかできないと思われる。したがって、自由発話の認識のための言語モデルには、非文を生成しないこと、そして perplexity が低いことが要求され、モデルがカバーできない範囲は garbage モデルや音素スキップで対処するのが妥当であると考えている。

5.4 音素スキップと garbage モデルの比較

今回の実験では、冗長語や言い直しを音素モデルでスキップする方法が garbage モデルより高い認識性能が得られた。これは、言語モデルが適応できない音声区間は音素モデルで対応したほうが garbage モデルで対応するより認識性能は高くなることを示している。しかし、この音素スキップの方法は garbage モデルより一般的に広いビーム幅が必要になると考えられる。したがって、語彙数が多い場合やビーム幅が小さい場合、garbage モデルのほうが認識性能は高くなる可能性があると思われる。

6 まとめ

本論文では、初めに連続音声認識アルゴリズムとしてフレーム同期型のフルサーチアルゴリズムについて述べた。次に計算量およびメモリ量を削減したアルゴリズムを提案した。これらは、ビームサーチの採用とその改良に依存している。

次に自由発話の認識に向けてアルゴリズムの改良を行った。自由発話では認識精度の高い音響モデ

ルを作成することは困難であると考えられる。そこで perplexity の低い言語モデルに着目し、単語の trigram を選択した。また、自由発話に特有な間投詞や言い誤りは、音声のあらゆる場所に出現する可能性がある。そこで、言語モデルにおいて、これらの言語現象をスキップすることで対処した。

最後に、このアルゴリズムを使用して自由発話の認識実験を行った。その結果、ビーム幅 16384 語彙数 435 において 47.7%の文認識率が得られた。そして意味的に正しい文を正解に含めた場合、1 位文理解率で約 75%、8 位までの累積文理解率は 90% になった。ただし、この自由発話のデータは、テキストを覚えて、その意図を理解し、自由に発話した音声データであるため、言語モデルに対して text-open のデータであるが、発話内容は text-closed に近い。 これらの実験の結果、このアルゴリズムの有効性が示された。

謝辞

この研究は著者が ATR 音声翻訳通信研究所に出向中におこなったものです。実験においては、HMM の不特定話者モデルは小坂哲夫氏 (現キヤノン) から頂きました。また、磯谷亮輔氏 (現 NEC) には deleted-interpolation の値や連続音声認識のアルゴリズムに関してコメントを頂きました。さらに音声翻訳通信研究所の第一研究室の方々には熱心な御討論と有益な御助言をいただきました。そして、音声翻訳通信研究所山崎泰弘社長および第一研究室匂坂室長には研究の機会を与えて頂きました。これらの皆様に感謝致します。

References

[1] Kai-Fu Lee, “Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System”, 15213 CMU-CS-88-148 (Apr. 1988).

[2] 村上仁一, 荒木哲朗, 池原悟, “日本文音節入力に対して 2 重マルコフ連鎖モデルを用いた漢字かな交じり文節候補の抽出精度” , 信学会論文誌, D-2, Vol.J75-D-2, No.1, pp.11-20 (Jan.1992).

[3] 村上仁一, 嵯峨山茂樹, “自由発話音声認識における音響的および言語的な問題点の検討”, 電子情報通信学会技術報告 SP91-100, pp.71-78 (Nov. 1991)

[4] 迫江博昭, 藤井浩美, 吉田和永, 亘理誠夫, “フレーム同期化、ビームサーチ、ベクトル量子化の統合による DP マッチングの高速化”, 信学論 D, Vol.J71-D, No.9, pp.1650-1659 (Sep. 1988).

[5] Douglas B. Paul, “New Developments in the Lincoln Stack-Decoder Based Large-Vocabulary CSR System”, Proceedings of the ARPA Spoken Language Systems Technology Workshop (1995).

[6] 徳永 豪, “ランダムアルゴリズムの話題から”, 電子情報通信学会誌, Vol.77, No9, pp.957-967 (1994-9).

[7] 伊藤 克亘, 速水 悟, 田中 穂積, “連続音声認識における未知語の扱い”, 電子情報通信学会技術報告, SP91-96 (1991-12).

[8] Higgins
A.L. and Wohlford R.E, ”Keyword recognition using template concatenation”, Proc.ICASSP85, pp. 1233-1236 (March 1985).

[9] Wilpon G., Rabiner R., Lee C. and Goldman R., “ Automatic Recognition of Keywods in Unconstrained Speech Using Hidden Markov Models”, IEEE Trans, ASSP-38,11 pp. 1870-1878 (1990).

[10] 井ノ上直己, 武田 一哉, 山本誠一, “Garbage HMM を用いた自由発話文中の不要語処理手法”, 電子情報通信学会論文誌, A Vol.J77-A, No.2, pp.215-222 (1994.2).

[11] A.Asadi, R.Schwartz and J.Makhoul, “ Automatic Modeling for Adding New Words to a Large Vocabulary Continuous Speech Recognition System”. ICASSP91 (1991).

[12] 北 研二, 江原 暉将, 森元 逞, “連続音声認識における未知語処理”, 日本音響学会講演論文集, 3-5-3, pp.93-94 (Mar. 1991)

[13] 甲斐 充彦, 中川 聖一, “日本語連続音声認識システム SPOJUS-SYNO の改良と評価”, 電子情報通信学会技術報告, SP93-20 pp.49-56 (1993-06).

[14] 南 泰浩, 山田智一, 鹿野清宏, “番号案内を対象とした大語彙連続音声認識アルゴリズム”, 電子情報通信学会技術速報, SP92-108 (Dec. 1992).

[15] F.Jelinek, “Self-Organized Language Modeling for Speech Recognition”, Readings in Speech Recognition, Morgan Kaufmann Publishers, Inc. San Mateo, California pp.450-506, (1990)

[16] 花沢利行, 中島邦男, “音声タイプライタを用いた未知語検出方法の改良検討”, 日本音響学会平成 4 年秋季研究発表会講演論文集, pp.219-220, (Oct. 1992).

[17] 江原 暉将, 小倉 健太郎, 森元 逞, “電話対話データベースの構築”, 第 40 回情報処理全国大会, pp.486-491 (1990).

[18] Cambridge University Engineering Department Speech Group and Entopic Research Laboratories Inc., “HTK:Hidden Markov Model Toolkit V1.5” (23 Sepember 1993).