修士学位論文

音響類似性に基づく隠れマルコフ網を用いた連続音声認識に関する研究

東北大学大学院工学研究科
電気及通信工学専攻
鈴木 基之

目 次

第 1 章 序論 1
1.1 研究の背景 1
1.1.1 音声認識システムの構成 1
1.1.2 Hidden Markov Model 2
1.1.3 音素認識部 4
1.1.4 言語処理部 6
1.2 研究の目的 7
1.3 本論文の構成 7

第 2 章 逐次状態分割法の概要 9
2.1 はじめに 9
2.2 逐次状態分割法の考案された背景 9
2.3 逐次状態分割法のアルゴリズム 10
2.4 逐次状態分割法の特徴 13
2.5 まとめ 14

第 3 章 逐次状態分割法の高速化 15
3.1 はじめに 15
3.2 逐次状態分割法による HMnet の学習速度 15
3.3 高速逐次状態分割法 16
3.4 HMnet の構成実験 18
3.5 まとめ 19

第 4 章 環境要因を必要としない音素 HMnet の構成法 20
4.1 はじめに 20
4.2 コンテキスト依存モデルの本質的問題点 20
4.3 与えた要因が不十分な場合の逐次状態分割法 21
4.4 環境要因を必要としない音素 HMnet の構成法 22
4.4.1 環境要因を必要としない音素 HMnet 構成法のアルゴリズム 22
4.4.2 環境要因を必要としない音素 HMnet 構成法の特徴 24
4.5 音素認識実験 24
4.5.1 6 子音の認識 25
4.5.2 全音素の認識 28
4.6 考察 30

4.6.1 環境要因を必要としない音素 HMnet の構造 30
4.6.2 音素別認識率 30
4.6.3 認識時の計算時間 33
4.6.4 学習サンプルの質による頑健性 33
4.7 まとめ 34

第 5 章 離散型 HMnet を用いた言語モデル **35**
5.1 はじめに 35
5.2 従来の統計的言語モデル 35
5.3 離散型逐次状態分割法 36
5.4 離散型 HMnet の性能評価 38
5.4.1 実験した手法 38
5.4.2 言語モデルの性能評価値 40
5.4.3 HMnet の構成実験 40
5.5 まとめ 43

第 6 章 結論 **44**
6.1 本研究の成果 44
6.2 今後の課題 45

謝辞 **47**

参考文献 **48**

研究業績 **50**

付録 A Baum-Welch の再推定アルゴリズム **51**
A.1 EM アルゴリズム 51
A.2 Baum-Welch アルゴリズム 51
A.3 HMnet のパラメータ推定 54

図目次

1.1 音素ラティスの例 2
1.2 left-to-right HMM 3
1.3 ergodic HMM 3
1.4 HMnet 4
1.5 本論文の構成 8

2.1 SSS のアルゴリズム 10
2.2 逐次状態分割法によって構成された HMnet の例 13

3.1 新しい状態に割り当てるガウス分布の計算 17
3.2 計算時間 19
3.3 認識率 19

4.1 与えた環境が不十分の場合の分割 22
4.2 “context table” の例 24
4.3 話者 FKN の 6 子音認識率 25
4.4 話者 FKS の 6 子音認識率 25
4.5 話者 FTK の 6 子音認識率 26
4.6 話者 FYM の 6 子音認識率 26
4.7 話者 MHO の 6 子音認識率 26
4.8 話者 MHT の 6 子音認識率 26
4.9 話者 MMY の 6 子音認識率 27
4.10 話者 MTK の 6 子音認識率 27
4.11 話者 MSH の 6 子音認識率 27
4.12 話者 MYI の 6 子音認識率 27
4.13 話者 FKN の全音素認識率 29
4.14 話者 FTK の全音素認識率 29
4.15 話者 MHO の全音素認識率 29
4.16 話者 MTK の全音素認識率 29
4.17 環境要因を必要としない音素 HMnet(音素 /b/ に対応) 31

5.1 離散型逐次状態分割法のアルゴリズム 37
5.2 DP パスにかかる重み 39
5.3 エディタ制御コマンドの文法 41
5.4 perplexity の比較 41

5.5 方法 2 で構成した HMnet 42

A.1 HMM の例 52
A.2 トレリスの例 52

表 目 次

3.1 混合分布についての計算量の比較 18
3.2 実験条件 18

4.1 6 子音の音素認識率 (状態数 110) 28
4.2 全音素での音素認識率 30
4.3 音素環境が通っている状態 (一部) 31
4.4 話者 MHO の音素別認識率 32
4.5 尤度計算を行なった平均パス数 33

5.1 実験した手法 40

第１章

序論

1.1 研究の背景

近年，各種 OA 機器の普及・発展にはめざましいものがあり，それにともなってより自然なマン・マシンインターフェースが望まれるようになってきた．特に音声は誰でも簡単，高速に意思を伝達できる手段として，インターフェースへの応用が強く期待されている．このような要望に応えて多くの音声認識に関する研究がなされており，現在では離散的な単語の認識装置が製品化され，その結果簡単な機械との対話が実現した．しかし，利用する側である人間に課せられた発声の制約 (例えば，発声様式，語彙数など) は少なくはなく，いまだ，知的水準に至っているとはいいがたい．このような背景のもと，連続音声の，とりわけ話し言葉のような自然発声 (spontaneous speech) を対象にした音声認識の研究に重点が置かれるようになった．そこで本論文では，連続音声を対象とした音声認識システムの高精度化を目標とする．まず本章では，現在の音声認識手法を簡単に解説し，その問題点を挙げる．

1.1.1 音声認識システムの構成

以下に従来からの代表的な音声認識システムの構成を示す．

1. 音響処理部
   入力された波形データを短区間 (通常 5ms から 10ms 程度) に切りわけ，短区間毎に音声の特徴をよく表現した特徴量へと変換する．現在最もよく用いられている特徴量は cepstrum である．cepstrum は人間の発声機構を考えて考案された特徴量であり，音声の特徴をよく表現している．

   人間は声帯で一定の励振波形を発生させ，声道の形を変えることで波形を変化させて様々な音声を発声する．つまり音声は声帯で発生された励振波形に声道の伝達関数を畳み込むことで生成される．cepstrum は，入力波形をフーリエ変換した後に対数をとり，更に逆フーリエ変換をして低次の項をとることで，声道の形からくる特徴をうまく取り出している．また，cepstrum の時間的な動きも特徴量にいれるために，cepstrum の時間方向の微係数も特徴量とするのが一般的である．

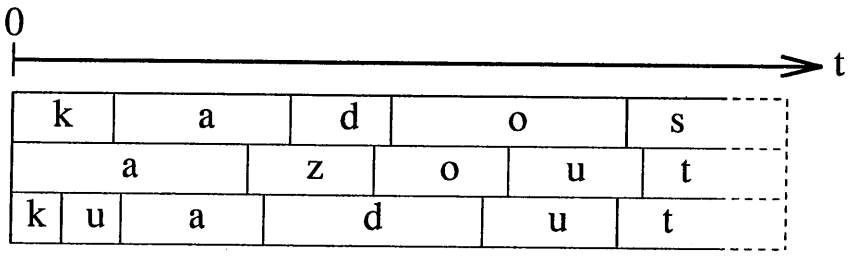


図 1.1: 音素ラティスの例

2. 音素認識部
   得られた特徴量ベクトル系列をもとに，音素の認識を行なう．あらかじめ各音素毎にその音素の特徴をよく表現するような音素モデルを用意しておき，入力音声と各モデルとの距離を計算，その距離が最も近い音素を認識結果とする．しかし，入力音声の音素区切りは未知のために，認識結果は図 1.1のような音素ラティスとなる．

3. 言語処理部
   音素ラティスから，単語辞書や文法辞書，言語モデルなどを使って文を生成する．まず音素ラティスから単語辞書を使って単語ラティスを生成し，文法的知識や言語モデルを使って文を生成する手法が多い．また，キーワードのみ検出すれば文の意味を理解できるという立場にたてば，word spotting という方式がとられる．

これらの処理は bottom-up 的であり，音素ラティスや単語ラティスなどの中間的なコードを生成しながら文の認識を行なう．ところが，人間が実際に音声を認識する時は，これとはかなり違った流れであることが予想される．つまり，人間は音声の一部が聞きとれなくても言語的知識 (単語や文法といった知識) や話の流れなどから無意識のうちに補間し，正確に認識をする．ある音声波形に対し，1 つの音素に対応する波形をノイズで置換したものを被験者に聞かせた結果，正確に音素を補間したという報告もある．つまり人間は，言語的制約などから音素認識部に情報を与えるなど，top-down 的な処理をしていると思われる．そこで最近では，言語モデルから次にくる音素を予測し，それに基づいて音素認識を行なうといった方式 [1] がとられ，よい性能を示すことが報告されている．これからの音声認識は，top-down 的な，あるいは言語知識といった高次の情報を音素認識部へとフィードバックさせたシステムが必要であると思われる．

1.1.2 Hidden Markov Model

現在よく用いられている音素認識の手法には，Hidden Markov Model(HMM) を用いる方法，DP によるテンプレートマッチング法，Neural Network を用いる方法などがある．この中で特に，HMM を用いる方法は，

- 発声のゆらぎなどを統計的に扱える
- 多量のサンプルから自動学習できる
- 音声とモデルの対応が比較的明確

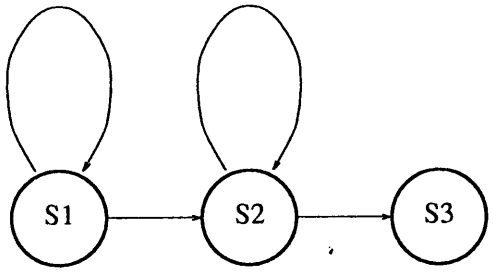


図 1.2: left-to-right HMM

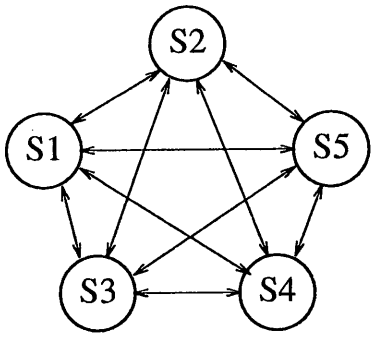


図 1.3: ergodic HMM

などの利点があり，音素認識の主流になっている．

ここで，HMM を用いての音素認識手法を簡単に説明する．まず，多量の学習データから音素毎に HMM を学習する．そのためには学習サンプルを音素毎に切り分ける必要があるが，音素区切りは一般に曖昧であることが多いため，発声内容のとおりに HMM を連結して学習する連結学習法 [2] も提案されている．学習アルゴリズムは最尤推定法である Baum-Welch アルゴリズム[1]がよく知られている．この方式は局所的最適値もしくは鞍点に収束することが証明されており，最適な収束値を得るために初期値が重要であることがいわれている．このようにして音素毎に学習した HMM を用意し，未知入力$\boldsymbol{X} = \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots \boldsymbol{x}_T$に対して，次のように認識する．各音素 HMM の最終状態から別の音素 HMM の初期状態への遷移確率 $Tr(r,p)$ を定義し，すべての音素 p について，式 (1.1)，式 (1.2)の漸化式にしたがって最大尤度をとる系列を計算する．

$$Q_p(1,1) = \log b_p(\boldsymbol{x}_1|1) \tag{1.1}$$

$$Q_p(j,t) = \left\{ \begin{array}{ll} \max\limits_i \{Q_p(i,t-1) + \log a_p(i,j)\} & j > 1 \\ \max \left\{ \begin{array}{l} \max\limits_r \{Q_r(E_r,t-1) + \log Tr(r,p)\} \\ \max\limits_i \{Q_p(i,t-1) + \log a_p(i,1)\} \end{array} \right. & j = 1 \end{array} \right\} + \log b_p(\boldsymbol{x}_t|j) \tag{1.2}$$

ここで，

$a_p(i,j)$	:	音素 p の HMM で状態 i から j への遷移確率
$b_p(\boldsymbol{x}_t\|j)$	:	音素 p の HMM で状態 j が t フレーム目の特徴量ベクトル$\boldsymbol{x}_t$を出力する確率
$Tr(r,p)$	:	音素 rから音素 p への遷移確率
E_p	:	音素 p の HMM の最終状態番号

ただし，各 HMM の最終状態 E_pは，次の音素への遷移のみ許されているものとする．最終的に $\max\limits_p Q_p(E_p,T)$ となる $Q_p(E_p,T)$ からバックポインタをたどることで，最適な音素系列を得る．

ここで，音素 HMM 間の遷移確率 $Tr(r,p)$ を日本語に現われる音素の接続に対して 1 に，それ以外は 0 に設定することで，音声タイプライタを構成することができる．また，$Tr(r,p)$ に言語モデルからの確率をいれることもできる．

HMM の形状としては，図 1.2 のような left-to-right 型を用いるのが一般的である．この型の利点は，前の状態へ戻る遷移がないために，時間的な変化をよく表現できる点である．ところが，

[1] 具体的な再推定式は付録 A参照.

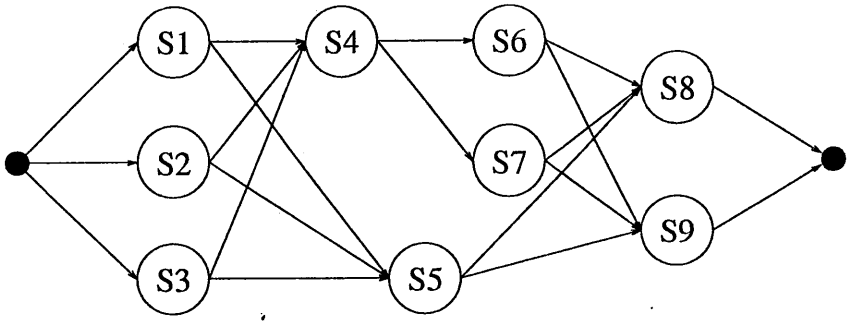


図 1.4: HMnet

すべてのサンプルが必ず同じ状態を遷移していくために，サンプル間の特徴量の差は状態が持っている出力分布で吸収するしかなく，あまりに特徴の違うサンプルを 1 つの HMM で表現するには無理が生じてしまう．これに対して，すべての状態間の遷移を許した ergodic HMM (図 1.3) と呼ばれる型を用いた方法 [3] も提案されている．この型は，サンプルの特徴によって通る状態が変わるので，left-to-right 型のように必要以上に出力分布が拡がることはない．しかし，HMM が小規模な場合は各状態が ergodic な結合をしているためにサンプルの時間的遷移をうまく表現できず，またそれを表現するために大規模なモデル構成にすると，パラメータの推定が大変になってしまい，あまり一般的ではない．

そこで，両者の中間的な型として，図 1.4のような隠れマルコフ網 (Hidden Markov Network: HMnet) が考えだされた．この型は，left-to-right HMM から見れば，数多くの HMM を似ている状態を共有させることで 1 つにまとめたもの，と見ることができ，また ergodic HMM から見れば，自分自身に戻ってくるような遷移を除いたもの，と見ることができる．つまり，時間的な遷移の表現能力を持ちながら，パスが並列に存在するために left-to-right 型のように必要以上に出力分布が拡がることもない．HMnet を構成するアルゴリズムの 1 つに逐次状態分割法 (Successive State Splitting:SSS) がある．このアルゴリズムは効率のよいコンテキスト依存モデル (1.1.3節で後述) を構成するために提案され，よい認識率を示すことが報告されている [4,5].

そこで，本研究では，この HMnet をモデルとして，連続音声の高精度な認識を目指す．音声認識の一般的なシステムは，音素認識部と言語処理部に分けられるが，本研究では，どちらの処理にもモデルとして HMnet を用い，それぞれの性能を向上させることを目的とする．

1.1.3 音素認識部

音声の認識単位は，音素 (phoneme) や音節，単語などが考えられるが，現在は音素を用いるのが一般的である．それは，比較的種類が少なく，また音素モデルを連結することで音韻や単語などのモデルを容易に構成できるからである．しかし，音素は発声された環境によって容易に音響的特徴が変形してしまう．その主な原因は，“音素が調音結合をおこす” ということである．つまり，/a/ のあとに発声された /k/ と，/o/ のあとに発声された /k/ とでは，同じ /k/ という音素でありながら，音響的にかなり変わった特性を持つ．このように，音素の音響的な特徴は，音素環境 (先行，後続音素など) や話者，発声速度などで容易に変形をうけてしまう．それらの変形をうけた音素サンプルをひとつの HMM で表現しようとすると，HMM の持つ出力確率分布の分散が拡がってしまい，特徴量空間で HMM がカバーする領域が他の HMM と重なるようになるので

誤認識の原因となる．

そこで，これらの問題を解決するために，音素を変形させる原因となっている環境毎にHMMを構成する方法が提案された．このHMMはコンテキスト依存モデル(context-dependent models)と呼ばれ，通常は音素を変形させる環境要因として先行音素と後続音素が与えられる．しかし，すべての先行音素と後続音素の組毎にHMMを構成しようとすると，HMMの数があまりに多すぎるためにそれに見あうだけの学習サンプルが集められず，現実的ではない[2]．そこで，各HMM間で音響的に似ている状態を共有化することが考えられた．これには大別して次の2つの手法がある．

- top-down 的な手法 [4,5]
  この手法は，小規模な初期HMMを用意し，1つの状態で表現するには無理のある部分を分割していくことで最終的なモデルを得ようとするものである．本論文で使用するHMnetの構成法である逐次状態分割法も，top-down 的な手法の1つである．逐次状態分割法では，小規模なHMnetから徐々に大規模なモデルが構成されるために，学習に使えるサンプル数にあわせた規模のHMnetが得られたところでアルゴリズムを止めることができる．しかし，逐次分割を繰り返してtree状にHMnetを構成していくために，似たような特徴を持つ状態が複数できる可能性があり，HMnetの効率という点では問題点が残る．

- bottom-up 的な手法 [6,7]
  top-down 的な手法に対して，過剰に分割されたHMMから似たような状態を融合していくことでモデルを構成しようというのがこの手法である．この手法では，top-down 的手法の欠点である，似た状態が複数生成されることもなく効率的なモデルが得られる．しかし，初期モデルとして過剰に細分化されたHMMを用意する必要があり，初期モデルは少数の学習サンプルで学習しなければならないので，学習サンプルの特徴に偏ったものになってしまう．その結果，得られるモデルも学習サンプルの特徴に偏っていないモデルであるとはいい難い．また，どこまで分割すれば“過剰に”分割したモデルになるのか，という問題も生じる．

これらどちらの方法でもコンテキストを考慮しないHMMに比べて認識率がよいという報告がされている．

ところで，このコンテキスト依存モデルには本質的な問題がある．それは，音素の変形要因の選び方である．コンテキスト依存モデルは与えられた変形要因について，すべての環境の組み合わせの数だけ構成される．実際には状態の共有化を行なってモデル数を減らすが，もし膨大な学習サンプルがあったとしても，与えられた変形要因についての組み合わせの数だけしかHMMが構成されない．一般には変形要因として先行音素と後続音素が与えられることが多いが，それで必要かつ十分かどうかは，確認されていない．つまり，/sima/ と発声した時の/m/ と，/hima/ と発声した時の/m/ は，同じ音響的特性を持つかどうかはわからない．変形要因として先行音素と後続音素を選んだコンテキスト依存モデルでは上記の2つは同じ/i-m+a/[3]というHMMで表現される．もし上記の/m/ が違う特性を持っていた場合，/i-m+a/ というHMMは音響的特徴の大きく違ったサンプルを1つのHMMで表現していることになり，出力分布が必要以上に拡がっ

[2] 日本語24音素の組み合わせは，原理的に $24^3 = 13824$ 種類あり，そのうち実際に日本語に表れる組み合わせは3000種類あまりである．

[3] 音素環境を/A-B+C/のように表現する．これは，先行音素が/A/，後続音素が/C/である音素/B/という意味である．

たHMMになることは避けられない．その結果，先々行音素(この場合は/s/と/h/)についてはコンテキスト非依存モデルとなるので，認識性能は低下してしまう．

つまり，音響的特徴をよく表現したコンテキスト依存モデルを構成するためには，音素の変形要因を事前に調査し，それらをすべて考慮して構成する必要がある．コンテキスト依存モデルは，本質的にこのような問題を含んでいる．

1.1.4 言語処理部

音素認識は，現在のレベルでは特定話者の発声で85%から90%程度とまだまだ低く，実用化するためには，言語的な知識に基づく制約が必要になる．理想的には音声タイプライタ的な，日本語として許される音素並びをすべて受理するような言語モデルを用いればよいのであるが，現在の音素認識レベルでは，そのような言語モデルでは制約が弱く認識率の面で音声認識システムとして使い物にならない．そこで，対象とするタスクを限定し，それにあわせた言語モデルを用いるのが普通である．

一般的に使われている言語モデルには，文法的な知識を与えるモデルと，統計的なモデルの2つがある．前者の代表的な例としては，有限状態オートマトンや文脈自由文法(Context Free Grammer: CFG)などがある．これらは，与える文法がよくタスクを表しているならばよいモデルとなる．特にCFGは拡張LRパーザと組み合わせることで，直接認識部を駆動する方法[1]が提案されており，よい言語モデルとして注目されている．しかし，これらの文法的モデルでは，文法を人が手で与える必要があるために，構成するのに大変な労力と時間がかかるのが大きな問題である．

これに対し統計的言語モデルでは，多量の学習サンプルから自動的に学習できるために労力は少なくてすむ．また，音素認識部がHMMのような確率的モデルであるので，言語モデルも確率的なモデルであると音響的尤度と言語的尤度の融合がしやすく，音素認識と言語処理を一体化できる可能性がある．しかし，よい文法を与えられたモデルに比べると一般に絞り込み能力に乏しく，また学習サンプル数に対して推定すべきパラメータ数が多い場合には，学習サンプルに依存したモデルになってしまう．

また，これらの中間的なモデルとして，知識を与えた文法モデルに確率を付ける方法もある．これは，与える文法がかなり制約力の弱いものであっても，学習サンプルから確率を推定することで制約力を強めるものである．代表的なものにCFGの書き換え規則に確率を付与したもの(Stochastic CFG)があり，CYK法やInside-Outsideアルゴリズム[4]などの書き換え確率の推定手法が提案されている．また，書き換え確率をergodic HMMで学習し，動的に変化させる方法[8]も提案され，強い絞り込み能力を持つことが報告されている．

本研究では，2つのアプローチのうち“自動的に学習できる”という利点に注目し，統計的なモデルについて研究を行なう．従来から研究されている統計的な言語モデルは，n-gramモデルと，ergodic HMMモデルが主流である．これらのモデルについて簡単に説明する．n-gramモデルは，対象言語を$n-1$重のマルコフ過程とみなし，$n-1$個の単語[5]が生成された時の次に生成される単語を学習サンプル中から数え上げ，その条件付き確率を計算する．この方法の大きな利点は，学習サンプルからの数え上げによって容易にモデルを構成することができることである．ergodic HMMを用いる方法は単語を発生する離散型ergodic HMMを学習させることで確率つ

[4] Inside-Outsideアルゴリズムは文法を与えない時も自動的にSCFGを獲得することができ，その意味では統計的言語モデルの部類に入る．しかし，計算量が膨大になるのでかなり小規模なモデルしか得ることはできない．

[5] 処理単位は単語に限らないが，ここでは処理単位は単語であるとして説明する．

きネットワーク文法を獲得する．しかし，計算量の問題から小規模なモデルしか得ることができない．そこで，状態数を逐次増加させていくことで計算量を削減した学習アルゴリズムも提案され [9]，bigram を越える性能が得られている．

しかしこれらのモデルはいずれも，モデルの規模が小さい時は時間的遷移の記述能力に乏しく，また，モデルの規模を大きくすると推定すべきパラメータが膨大になり，学習が現実的ではなくなってしまう．そこで，これらに変わる言語モデルを開発する必要がある．

1.2 研究の目的

前節で述べたように，音素認識部や言語処理部の構成には HMnet が優れていると思われる．そこで，HMnet というモデルを使用し，連続音声認識の高精度化を目指す．HMnet の構成アルゴリズムは逐次状態分割法を基本とし，音素認識部と言語処理部のそれぞれについて，性能の向上をはかる．まず音素認識部では，コンテキスト依存モデルの本質的問題点である“考慮すべき環境要因を与える”点を解決するために，環境要因を必要としないモデルの構成法を提案する．また，言語処理部では，HMnet を言語モデルに適用するために，離散分布型の HMnet 構成法を提案する．

1.3 本論文の構成

本論文の構成は図 1.5のとおりである．

第 1 章 序論であり，研究の背景，及び本研究の目的を述べる．

第 2 章 HMnet の構成法である逐次状態分割法のアルゴリズムを述べ，その特徴について議論する．

第 3 章 逐次状態分割法の問題点の 1 つである“学習速度が遅い”点を改善するため，高速な逐次状態分割法を提案し，その有効性を示す．

第 4 章 コンテキスト依存モデルの本質的問題点を改善するため，環境要因を必要としない音素 HMnet の構成法を提案し，その有効性を示す．

第 5 章 離散型の HMnet を構成する逐次状態分割法を提案し，言語モデルとして応用する．

第 6 章 結論であり，本研究の成果と今後の課題について述べる．

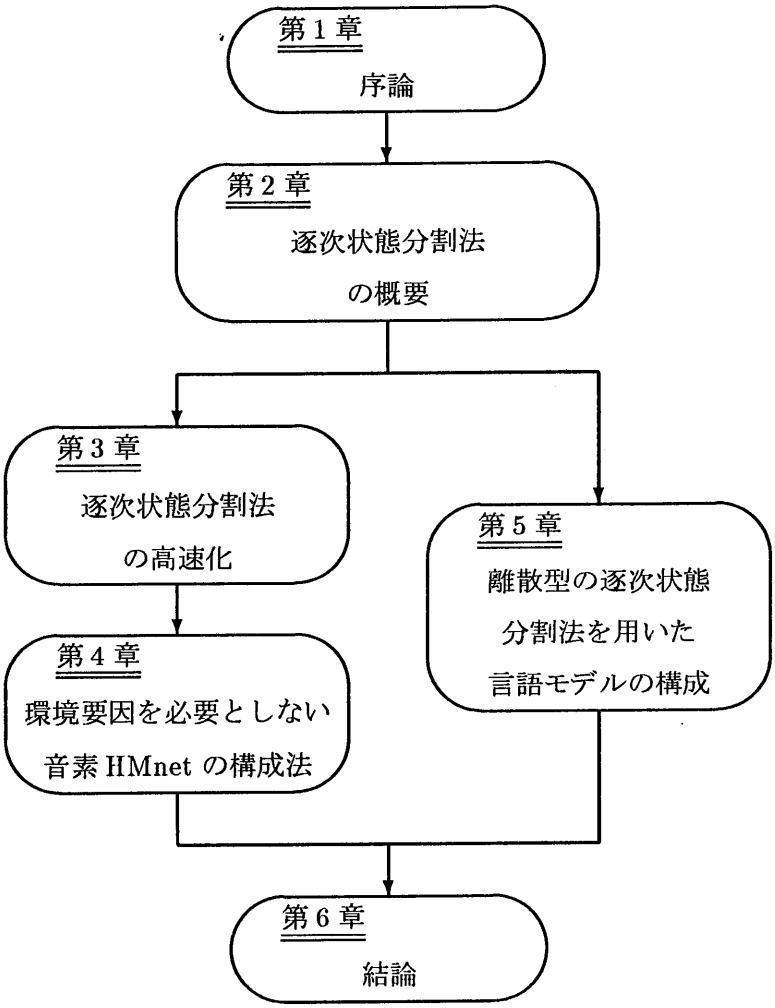

第 1 章
序論
第 2 章
逐次状態分割法
の概要
第 3 章
逐次状態分割法
の高速化
第 4 章
環境要因を必要としない
音素 HMnet の構成法
第 5 章
離散型の逐次状態
分割法を用いた
言語モデルの構成
第 6 章
結論

図 1.5: 本論文の構成

第 2 章

逐次状態分割法の概要

2.1 はじめに

HMnet を構成するアルゴリズムの 1 つに，逐次状態分割法 (Successive State Splitting: SSS) [4,5] がある．このアルゴリズムは，もともとコンテキスト依存モデルを HMnet で構成するために提案されたアルゴリズムであり，状態数を逐次的に増加させていくことで HMnet を構成する．その時の評価基準は，尤度最大という認識時と同じ評価基準であるため，よりよい認識率が得られることが期待される．

この章では，逐次状態分割法のアルゴリズムを説明し，その特徴と問題点を指摘する．

2.2 逐次状態分割法の考案された背景

最近の音声認識手法は，認識単位として音素を用いることが多い．しかし，音素は調音結合などによる変形が大きいため，1 つのモデルで表現するには限界がある．そこで，音素の音響的な性質を変形させる要因[1] まで考慮した，いわゆる異音 (allophone) を認識単位とする方法が試みられ，その有効性が認められつつある．

しかし，異音を認識単位とする場合には音素を認識単位とする場合に比べてモデル数が大幅に増加するため，学習サンプル数に制限がある場合のモデル学習が 1 つの大きな問題となっている．信頼性の高いモデルを構成するためには音素環境をうまくクラスタリングし，1 モデル当りの学習サンプル数を過度に減少させることがないよう工夫する必要がある．この認識単位を適切に設定する手法として，人が音声学的な知見に基づいて与える方法や，与えられたサンプルに対する歪み計算により，音素環境空間を分割していく方法，すべての異音モデルを学習したあとで，音響的に類似する状態を共有化していく方法などが提案されている．

しかし，これらの手法はいずれも先験的知識に基づいたり，認識時の尺度 (サンプルがモデルから生成される尤度) とは別の尺度によって構造が決定されたりしており，認識率が最大になるかどうかは保証されていない．また，通常の HMM では，状態をいくつ連結するか，といった構造決定も経験的な知識に基づいて決定されている．これらの問題の解決策としてモデルの構造を自動的に決定するアルゴリズムが逐次状態分割法である．

[1] ほとんどの場合，先行音素と後続音素とされる．

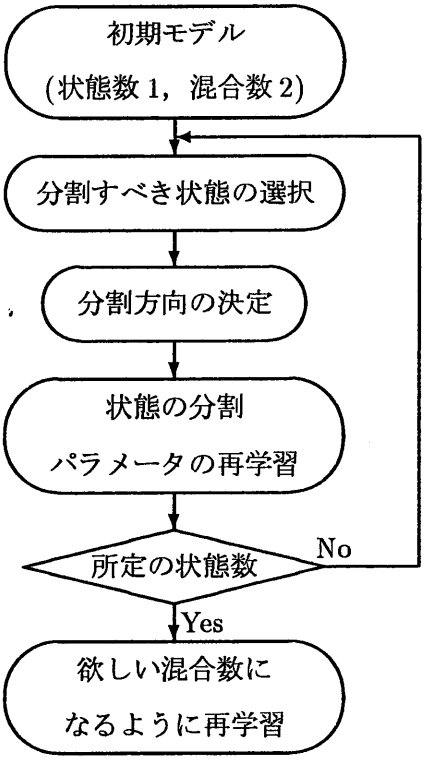


図 2.1: SSS のアルゴリズム

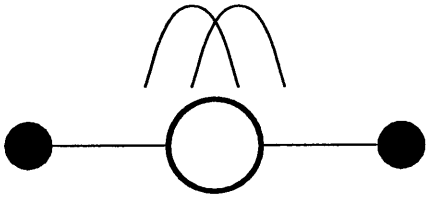
HMM によるコンテキスト依存モデルでは，すべての異音 (もしくは，適当にクラスタリングされたもの) について別々にモデルを構成する．この時，例えば/a-k+a/ を表現するモデルと，/a-k+i/を表現するモデルの前半部分は似ていることが予想される．その似ている部分の状態を共有化すれば，その状態の学習サンプルは増加したことになり，統計的により頑健な学習が可能となる．逐次状態分割法では HMnet をモデルとすることで，このような状態を共有した形のモデルを構成することができる．

2.3　逐次状態分割法のアルゴリズム

逐次状態分割法のアルゴリズムは図 2.1のようになる．各状態は，出力確率分布と遷移確率の他に考慮する環境要因毎に受理コンテキストのリストを持つ．

Step 1 初期モデルの学習

初期モデルとして，1 状態で出力分布として 2 混合のガウス分布 (対角共分散行列) を持つ HMM を用意し，すべての学習サンプルを使って学習する．各環境要因の受理コンテキストリストにすべての存在するコンテキストを加える．



Step 2 分割すべき状態の決定

すべての状態の中で，出力分布が最も拡がった状態を選び，分割すべき状態とする．式(2.1)で示される値 d_iは，2 混合のガウス分布を単一ガウス分布で近似した時の分散に相当する値に，その状態を推定するのに使われたサンプル数をも考慮した値となっており，統計的な頑健性の向上が図られている．

$$d_i = n_i \times \sum_k^K \frac{\sigma_{ik}^2}{\sigma_{Tk}^2} \tag{2.1}$$

$$\sigma_{ik}^2 = \lambda_{i1}\sigma_{i1k}^2 + \lambda_{i2}\sigma_{i2k}^2 + \lambda_{i1}\lambda_{i2}(\mu_{i1k} - \mu_{i2k})^2$$

ここで，

K : パラメータ次数

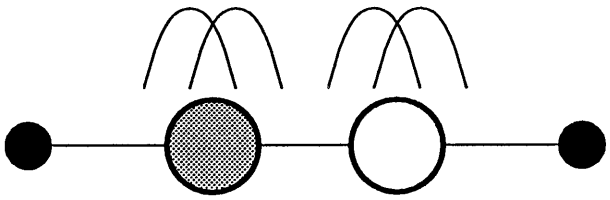
$\lambda_{i1}, \lambda_{i2}$: 状態 i の 2 つの分布の重み係数

μ_{i1k}, μ_{i2k} : 状態 i の 2 つの分布の平均

$\sigma_{i1k}^2, \sigma_{i2k}^2$: 状態 i の 2 つの分布の分散

n_i : 状態 i の推定に用いたサンプル数

σ_{Tk}^2 : 全サンプルの分散 (正規化係数)

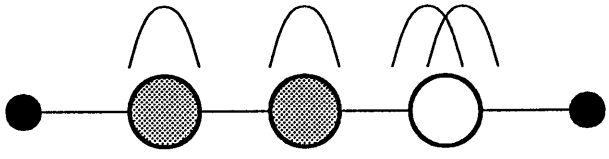


Step 3 状態の分割

Step 2 で決定された状態を 2 つに分割する．この時，新しい状態の出力確率分布は，分割された状態が持っていた 2 混合のガウス分布を 1 つずつ割り当てる．その後，新しい状態の配置を時間方向 (直列) に連結した場合の学習サンプルに対する尤度 P_t と，コンテキスト方向 (並列) に連結した場合の尤度 P_c を計算し，より尤度の高い方を採用する．P_t と P_cは，以下のようにして計算される．

- 時間方向への分割

  時間方向へ分割する時は，どちらの状態を先に置くかで 2 とおりの可能性がある．そこで，2 つの可能性についてそれぞれ尤度を計算し，その高い方を P_t とする．

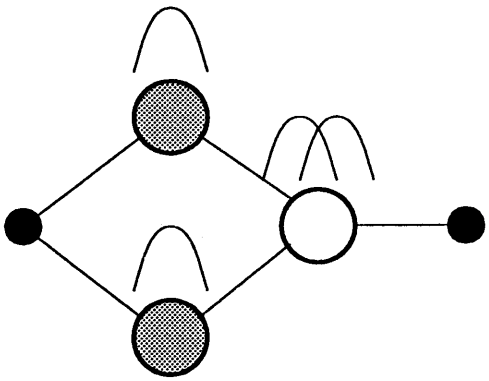


- コンテキスト方向への分割
  コンテキスト方向への分割は，パスが 2 つに別れるためにそれぞれの学習サンプルがどちらの状態を通るかを決定する必要がある．そこで，ある環境要因 (先行音素など) について学習サンプルをその要因の要素ごとにまとめ，その集合毎に尤度の高い方の状態を通るようにする．

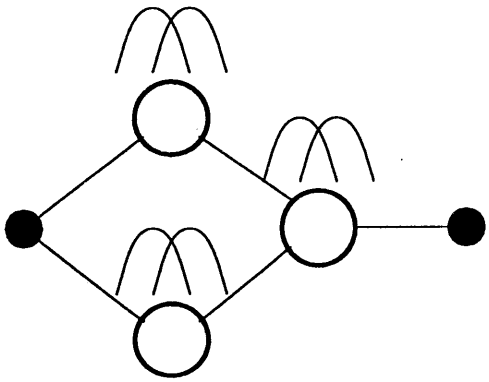
$$P_c = \max_j \sum_l \max(P_m(y_{jl}), P_M(y_{jl})) \tag{2.2}$$

ここで，

j : この状態において分割可能な要因
y_{jl} : 要因 jの値が l番目の要素である学習サンプルの部分集合
$P_m(y_{jl})$: y_{jl}を状態 m に割り当てた時の尤度
$P_M(y_{jl})$: y_{jl}を状態 Mに割り当てた時の尤度



Step 4 分布の再推定
この時点で新しい状態には，単一ガウス分布が割り当てられたままになっている．そこで，すべての状態が 2 混合のガウス分布を持つように，HMnet 全体を再学習する．その後，所定の状態数になるまで，Step2, Step3 を繰り返す．



Step 5 分布の変更
これまでの処理で HMnet の形状が決定される．そこで最後に，各状態に割り当てられている出力確率分布を実際に使用したい混合数になるように HMnet 全体を再学習する．

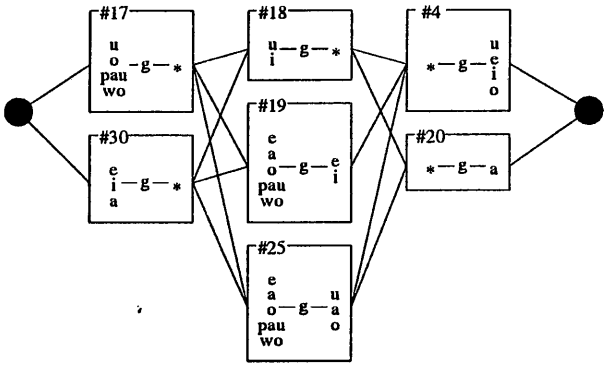


図 2.2: 逐次状態分割法によって構成された HMnet の例

このようにして得られた HMnet の例を図 2.2に示す．これは，HMnet のなかの音素 /g/ に対応する部分だけ抜き出したものである．図 2.2に示されているように，各状態は受理する先行音素と後続音素のリストを持つ[2]．そこで，音素の認識時に認識すべき音素の音素環境を言語的モデルなどから予測して与えることで，その音素環境に対応するパスがただ 1 本に決まる．例えば，言語モデルなどから先行音素が /a/ 後続音素が /e/ であると予測された時，音素 /g/ に対応するモデルは，#30 – #19 – #4 というパスになり，通常の HMM とみなすことができる．逐次状態分割法で構成された HMnet は，このようにしてパスを 1 本に制限することでコンテキスト依存モデルのように使うことができる．

また，HMnet の形状を見ると，前半の状態 (#17 や#30) は先行音素について状態が分割され，また後半の状態 (#4 や#20) では後続音素について分割されている．これは，音素の前半部分は先行音素から，また後半部分は後続音素からの影響が強いと予測されることと合致しており，音響的な特徴をよく表現しているものと思われる．

2.4　逐次状態分割法の特徴

逐次状態分割法の利点には，以下のようなものがある．

- 時間方向も含めて構造を自動決定できる．
  従来の HMM では，状態をいくつ連結するか，といった問題は人が経験的な知識から与えるものであった．また，コンテキスト依存モデルでパラメータを削減するためのモデルの共有なども，どの環境を共有するかということは人が与えたり，認識とは別の尺度 (例えば歪み最小など) でクラスタリングしたりするものが多かった．それを，尤度最大という認識と同じ尺度で自動決定できるところが，逐次状態分割法の最大の利点である．

- 環境の補間作用がある．
  コンテキスト依存モデルを構成する時，その種類があまりにも多いために学習サンプルにすべての組み合わせがあるとは限らない．このような場合でも，逐次状態分割法では環境の補間作用があるために学習サンプルに現れなかったコンテキストに対応するモデルが得られ

[2] “*” は，すべての音素を受理することを示す．

ることが期待される[3]．これは，すべての音素環境の直積空間を分割しながら HMnet を構成しているためで，例えば /a-k+a/ という環境が現れなかった場合，/a-k+i/ の前半部分と，/i-k+a/ の後半部分をとってきて，/a-k+a/ のモデルとする，といった原理で補間している．

ただし，アルゴリズムの性質上，逐次状態分割法特有の問題点もある．

- 与えた環境要因が不十分であると，分割ができなくなる．
- 学習の計算時間が，通常の HMM に比べてかなり遅い．

与える環境要因が不十分である時は，一般に音響的特徴をよく表現したコンテキスト依存モデルは構成されない．特に，逐次状態分割法の場合は構成の途中でアルゴリズムが止まってしまうことが起こる．これについては 4.3節でくわしく述べる．

2.5　まとめ

HMnet の効率的な構成法である逐次状態分割法のアルゴリズムを述べ，その特徴を議論した．逐次状態分割法はもともとコンテキスト依存モデルを効率よく構成する目的で提案されたものであるので，この方法で構成した HMnet をそのまま環境要因を与えない音響モデルや，言語モデルとして適用することはできない．そこで，次章以降で逐次状態分割法を修正し，本論文の目的に合った HMnet を構成するようにする．

[3] 必ず得られる，というわけではない．

第 3 章

逐次状態分割法の高速化

3.1 はじめに

逐次状態分割法には，“学習速度が通常の HMM に比べて非常に遅い” という大きな欠点がある．これは，状態を 1 つ増やすたびに HMnet を再学習するためで，計算機の能力が飛躍的に向上した現在でも不満が残るほど遅い[1]．そこで，逐次状態分割法のアルゴリズムを改良し，高速に HMnet を構成する方法を提案する．このアルゴリズムは原理的にオリジナルとほぼ同じ動作をするので，同等の性能を持つ HMnet が得られることが期待される．

3.2 逐次状態分割法による HMnet の学習速度

逐次状態分割法による HMnet の学習が通常の HMM に比べて非常に遅い原因として，以下のようなことが考えられる．

- 状態が 1 つ増えるたびに HMnet 全体を再学習する必要がある
- 各状態が 2 混合のガウス分布を持つ

このうち，2 混合のガウス分布を持つ HMM は，単一ガウス分布を持つ HMM に比べて一般に学習に時間がかかることが知られている．それは，以下のような理由による．

- 初期値をクラスタリングによって設定する必要がある
  Baum-Welch の再推定アルゴリズムは局所的最適値もしくは鞍点に収束する．そこで最適な収束値を得るためには初期値の設定が重要になってくる．通常単一ガウス分布を持つ HMM ではその状態を通る全学習サンプルの平均値を初期値とするが，混合ガウス分布を持つ HMM の場合はそれらのサンプルを混合数分にクラスタリングし，それぞれのクラスタ中心を初期値とする方法がとられる．
- Baum-Welch の再推定アルゴリズムの収束が，単一ガウス分布の HMM に比べて遅い
  これは経験的にしか議論できないが，同じ収束条件のもとでは 2 混合のガウス分布を持つ HMM の収束は，単一ガウス分布を持つ HMM に比べて数倍から数十倍遅い．

[1] SPARCstation 2 で計算した場合，比較的学習サンプルの少ない 6 子音の実験で状態数 80 まで分割するのに 18 時間程度である．

本章では，アルゴリズムを原理的に変えない方向での高速化を図るために，2 混合のガウス分布ではなく，主に単一ガウス分布を使う逐次状態分割法を提案する．

3.3　高速逐次状態分割法

オリジナルの逐次状態分割法で 2 混合のガウス分布を使っていた理由は，状態の分割時 (2.3節のアルゴリズム Step 3) に，分割してできた新しい状態にガウス分布を 1 つずつ割り当てるためである．そこで，ここの部分を改良し，各状態が単一ガウス分布を持つような逐次状態分割法 [10] を提案する．

高速逐次状態分割法のアルゴリズムを以下に示す．なお，主な変更点は HMnet の各状態が単一ガウス分布を持つことによる変更と，Step 3 の状態の分割の部分である．

Step 1 初期モデルの学習

初期モデルとして，1 状態で出力分布として単一ガウス分布 (対角共分散行列) を持つ HMM を用意し，すべての学習サンプルを使って学習する．各環境要因の受理コンテキストリストにすべての存在するコンテキストを加える．

Step 2 分割すべき状態の決定

すべての状態の中で，出力分布が最も拡がった状態を選び，分割すべき状態とする．オリジナルでは，混合分布を用いていたために各パラメータから分布の拡がりを計算していたが，ここでは出力分布は単一ガウス分布であるので，その分散の値そのものに，推定に用いたサンプル数を乗じたものを基準とする．

$$d_i = n_i \times \sum_k^K \frac{\sigma_{ik}^2}{\sigma_{Tk}^2} \tag{3.1}$$

ここで，

K : パラメータ次数
σ_{ik}^2 : 状態 i の出力分布の分散
n_i : 状態 i の推定に用いたサンプル数
σ_{Tk}^2 : 全サンプルの分散 (正規化係数)

Step 3 状態の分割

Step 2 で決定された状態を 2 つに分割する．この時，新しい状態の出力確率分布を以下のようにして求める (図 3.1)．

Step 3-1 分割すべき状態を通るすべての学習サンプルについて，Viterbi アルゴリズムを使ってこの状態が出力するサンプルの部分系列を切り出してくる．

Step 3-2 Step 3-1 で切り出されたすべての学習サンプルの部分系列を用いて，1 状態，2 混合の HMM を学習する．

Step 3-3 得られた 2 つのガウス分布をそれぞれ新しい状態に割り当てる．

このようにして新しい状態の出力確率分布を求めた後，新しい状態の配置を時間方向 (直列) に連結した場合の学習サンプルに対する尤度 P_t と，コンテキスト方向 (並列) に連結した場

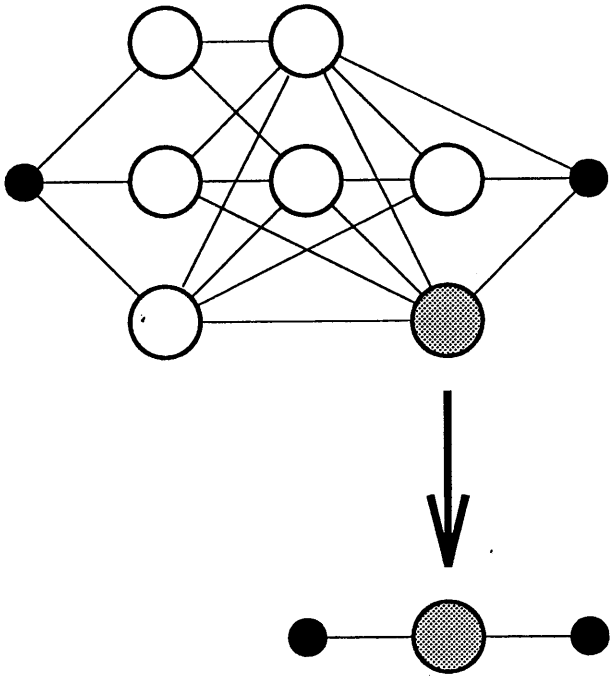


図 3.1: 新しい状態に割り当てるガウス分布の計算

合の尤度 P_cとを計算し，より尤度の高い方を採用する．P_tと P_cの計算方法はオリジナルと同様である．

Step 4 分布の再推定
分割終了後の最適なパラメータを求めるために HMnet 全体を再学習する．その後，所定の状態数になるまで，Step2, Step3 を繰り返す．

Step 5 分布の変更
これまでの処理で HMnet の形状が決定される．そこで最後に，各状態に割り当てられている出力確率分布を実際に使用したい混合数になるように HMnet 全体を再学習する．むろん，単一ガウス分布で使用したい時[2]はこの Step は省略できる．

このアルゴリズムを用いると，HMnet 全体は各状態が単一ガウス分布を持つために高速な学習が行なわれる．2 つのアルゴリズムの混合分布についての計算量は 1 サイクル (状態が 1 つ増える) あたり表 3.1のようになる[3]．HMnet の規模が大きくなればなるほど HMnet 全体のパラメータの再推定は時間がかかり，また学習サンプル数が増えれば増えるほど初期値設定のためのクラスタリングに時間がかかるので，そういった場合に高速化の効果がより一層表れると思われる．

また認識性能については，本来 Baum-Welch アルゴリズムで学習されるべきところを Viterbi アルゴリズムで系列を切り出して学習するという近似を行なっている．しかし，Viterbi アルゴリズムでの学習は Baum-Welch アルゴリズムでの学習に対して性能はほとんど落ちないことが知ら

[2]実際には，計算量などの問題から単一ガウス分布で使用する場合が多い．
[3]本章では，これ以降オリジナルの逐次状態分割法を SSS-original，提案した高速逐次状態分割法を SSS-fast と表記する．

表 3.1: 混合分布についての計算量の比較

アルゴリズム	SSS-original	SSS-fast
初期値の設定	2 状態	1 状態
パラメータの推定	HMnet 全体	1 状態の HMM

れており，結局 SSS-original で構成したものと同程度の性能を持った HMnet が得られることが期待される．

3.4　HMnet の構成実験

SSS-fast の学習時間と認識性能を評価するために，特定話者の音素認識実験を行なった．実験条件は表 3.2のとおりである．今回はおおまかな性能を見ればよいので，全音素での認識実験は行なわず，/b, d, g, m, n, N/ の 6 子音についてのみ行なった．音声データは ATR 連続音声データベース 503 文章中の 400 文章から，ATR が提供するラベルに従って切り出した音素を使った．また評価データは，学習に用いなかった 103 文章中の音素を使った．

Sun SPARCstation2 での計算時間を図 3.2に，認識率を図 3.3に示す．これを見ると，認識率はどちらも同程度であることがわかる．SSS-fast の方が多少認識率がよいように見えるが，この差に意味があるのか，それとも初期値の乱数によるゆらぎにすぎないのかはわからない．また，実際に得られた HMnet の形状も多少違っていたが，認識率を見る限り同等の認識性能を有しており，問題はないと思われる．

これに対し，計算時間にはかなりの差がみられる．状態数 80 で約 5 倍のひらきがある．この差は状態数の増加とともにひらいていく傾向にあり，HMnet が大規模になればなるほど提案手法による高速化の効果が現われてくると思われる．また計算時間の変化は，SSS-fast では直線状に，SSS-original では放物線状になっているように見えるが，計算時間のオーダがそれぞれ $O(N), O(N^2)$ (N は HMnet の状態数) に従うのかどうかはわからない．というのも，2 つのアルゴリズムの計算

表 3.2: 実験条件

認識タスク	/b, d, g, m, n, N/
話者	男性 1 名 (MMY)
パラメータ	logpow,cep(16),Δlogpow
	Δcep(16) からなる
	34 次元ベクトル
分析条件	サンプリング周波数 12kHz
	16bit 量子化
	20ms ハミング窓
	フレーム周期 5ms
学習サンプル	400 文章
テストサンプル	学習サンプル以外の 103 文章

時間は初期値の設定のためのクラスタリングの収束速度と学習アルゴリズムの収束速度に依存しているが，これらは，ともに理論的なオーダの見積りが難しいからである．しかし，仮に 2 つのアルゴリズムの計算時間が高々定数倍であったとしても，現実に HMnet を構成しようとした時の計算時間の差は大きく，このアルゴリズムは非常に有効であると思われる．

3.5　まとめ

逐次状態分割法を改良し，単一ガウス分布をベースとする高速逐次状態分割法を提案した．この方法は原理的にオリジナルと同等の動作をするため，オリジナルの逐次状態分割法で構成した HMnet と同等の性能を持つ HMnet を高速に構成することができる．

日本語 6 子音の音素認識実験では，認識率は同程度で，計算時間は状態数 80 まで分割した時に 1/5 程度まで削減できた．これは，もっと状態数を増やした時や，母音などのように学習サンプル数が多い時は，更に効果的になると思われる．

また，更なる高速化手法として，Step 4 を省略することも考えられる．これは，状態が分割されたことによるまわりの状態の出力分布への影響を無視したことになるので，多少の性能低下がみられると思われるが，HMnet 全体の再学習を行なわないためかなりの高速化になると思われる．このようにして構成された HMnet の認識性能と学習時間は，興味のあるところである．

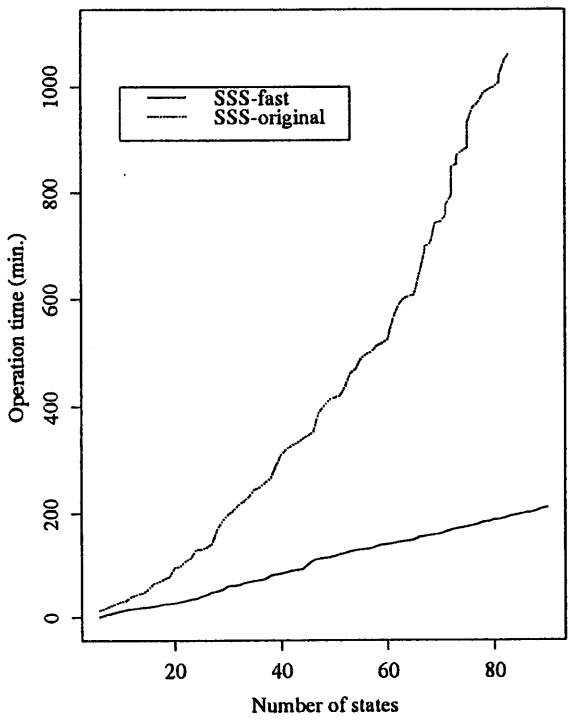


図 3.2: 計算時間

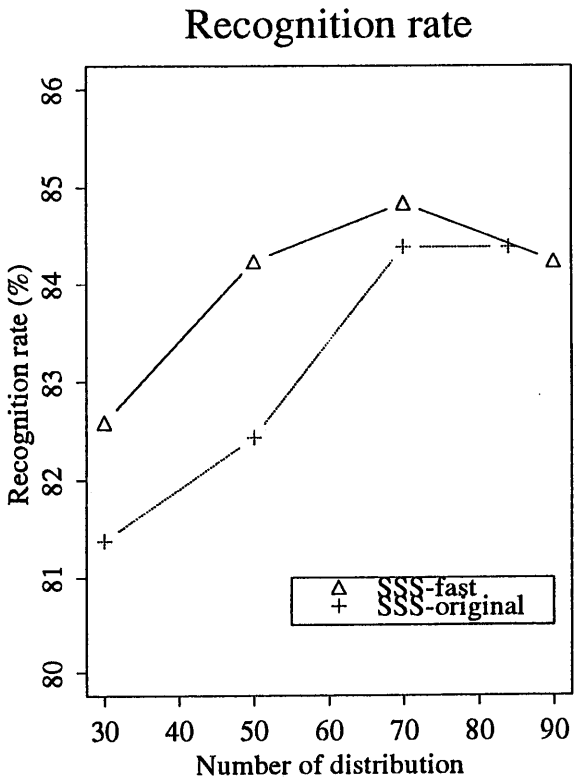


図 3.3: 認識率

第 4 章

環境要因を必要としない音素 HMnet の構成法

4.1 はじめに

一般的な認識単位である音素は，発声された環境 (話者，先行音素，後続音素，発声速度など) によって容易に変形し，それが誤認識の原因の 1 つとされている．この問題を解決するために音素モデルとして HMM を用いることが多いが，あまりに音響的特徴の違うものを 1 つの HMM で表現してしまうと，HMM の持つ出力確率密度分布の分散が拡がってしまう．その結果，特徴量空間内で他の HMM がカバーする領域との重なりが大きくなり，誤認識の原因となる．音素の変形を吸収する問題は，特に音素環境による変形を吸収する問題と，話者による変形を吸収する問題が注目され，それぞれ様々な解決策が提案されている．本章では，音素環境による音素の変形をうまく吸収することを目的とする．

4.2 コンテキスト依存モデルの本質的問題点

音素環境による音素の変形を吸収するために，音素環境毎に HMM を構成する方法がよく知られている．こうしてできた HMM をコンテキスト依存モデルと呼び，コンテキスト非依存なモデルに比べてよい認識率を示すことが報告されている．また，様々なコンテキスト依存モデル構成法 [4–7,11,12] が提案されているが，これらはすべて効率のよいコンテキスト依存モデルを構成するための手法であり，“音素環境毎にモデルを構成する” という基本姿勢はみな同じである．

すべてのコンテキスト依存モデル構成法は，前もって考慮すべき音素の変形要因を与えることが必要である．コンテキスト依存モデルは与えられた変形要因について，すべての環境の組み合わせの数だけ構成される．実際には状態の共有化や音素環境のクラスタリングなどを行なってモデル数を減らすが，もし膨大な学習サンプルがあったとしても，与えられた変形要因についての組み合わせの数だけしか HMM が構成されない．一般的には音素の変形要因として先行音素と後続音素が使われることが多いが，それで必要かつ十分かどうかは，確認されていない．つまり，/sima/ と発声した時の/m/ と，/hima/ と発声した時の/m/ は，同じ音響的特性を持つかどうかはわからない．変形要因として先行音素と後続音素を選んだコンテキスト依存モデルでは上記の 2 つは同じ/i-m+a/ という HMM で表現されるが，もし上記の/m/ が違う特性を持ってい

た場合，/i-m+a/ という HMM は音響的特徴の大きく違ったサンプルを 1 つの HMM で表現していることになり，出力分布が必要以上に拡がった HMM になることは避けられない．その結果，先々行音素 (この場合は/s/と/h/) についてはコンテキスト非依存モデルとなるので，特徴量空間で HMM がカバーする領域が他の HMM と重なるようになり，認識性能が低下するのは容易に想像することができる．つまり，音響的特徴をよく表現したコンテキスト依存モデルを構成するためには，音素の変形要因を事前に調査し，それらをすべて考慮して構成する必要がある．つまり，上記/sima/ , /hima/ の例で言えば，先々行音素/s/, /h/ が音素の音響的特徴を変形させている要因の 1 つであることをなんらかの方法で事前に調査し，先々行音素，先行音素，後続音素のすべての組み合わせ毎にモデルを構成する必要がある．

一方，環境要因を必要以上に与えた場合は，音素の変形に関与しているすべての環境毎にモデルが構成されるので，上記のようなモデルが拡がってしまうという問題は起きない．しかし，要因を与えれば与えるほど構成すべきモデルの数は指数関数的に増加し，多量の学習サンプルがあったとしてもその学習は現実的ではない．

結論として，よいコンテキスト依存モデルを構成するためには，音素の変形要因を過不足なく与える必要がある．しかし，適切な環境要因を調査するのは困難であり，また音素環境以外の要因 (例えば発声速度など) による音素の変形があった場合，それらをすべて網羅するように環境要因を与えるのは非常に難しい．結局，天下り的に“先行音素と後続音素” と決めてしまうのが現状である．コンテキスト依存モデルには，このような本質的問題点が存在する．

4.3　与えた要因が不十分な場合の逐次状態分割法

逐次状態分割法で構成された HMnet もコンテキスト依存モデルの 1 つなので，前節で述べたように，与えた環境要因が適切でなければ音響的特徴をよく表現したモデルにはならない．特に逐次状態分割法の場合は，与えた環境要因が不十分であるとアルゴリズムが止まってしまうことが起こる．ここでは，与えた要因が不十分な時にどのようなことが起こるかを説明する．

逐次状態分割法は前もって与えた環境要因毎に尤度を計算し，最も高い尤度を示すように要因を新しい状態へと振り分ける．ここで，図 4.1の例を考えてみる．いま，分割すべき状態を /a-m+e/, /a-m+o/ の 2 つの音素環境の学習サンプルが通っているとする．また，これらのサンプルは先々行音素が /k/ のものと /m/ のものとがあり，その違いによって音響的な変形を受けているものとする．つまり，音響的には先々行音素が /k/ である /a-m+e/, /a-m+o/ と，先々行音素が /m/ である /a-m+e/, /a-m+o/ という 2 つのグループにクラスタリングされる．この時，当然この状態が持つ 2 混合のガウス分布は，先々行音素が /k/ であるグループの音響的特徴を表現している分布 (分布 1 とする) と，先々行音素が /m/ であるグループの音響的特徴を表現している分布 (分布 2 とする) からなっている．さて，このような状況にある状態を後続音素について分割することを考える[1]．分割は，各音素環境がどちらの分布でより高い尤度を出すかで決定されるわけだが，図 4.1で示すように/a-m+e/ という音素環境である学習サンプルのうち先々行音素が /k/ であるものは分布 1 のほうが尤度が高く，また先々行音素が /m/ であるものは分布 2 のほうが尤度が高い．/a-m+e/ という音素環境としての尤度はこれらの和になるので，この場合は分布 1 を選択することになる．ところが，/a-m+o/ という音素環境であるサンプルも同様にして分布 1 を選択してしまうため，分割ができなくなってしまう．この原因は明らかに環境要因として先々行音素を与えなかったことにある．つまり，逐次状態分割法では音素の変形に影響

[1]先々行音素は環境要因として与えられていないため，先々行音素について分割することはできない．

a-m+e

先々行音素	分布 1	分布 2
k	231.58	74.05
m	83.74	195.30
	315.32	269.35

a-m+o

先々行音素	分布 1	分布 2
k	257.35	47.52
m	97.82	184.50
	355.17	232.02

(数字は各分布での尤度)

図 4.1: 与えた環境が不十分の場合の分割

している要因はすべて前もって与えておく必要がある.

4.4　環境要因を必要としない音素 HMnet の構成法

コンテキスト依存モデルの本質的問題を解決するために，考慮すべき環境要因を与えないモデル構成法 [13,14] を提案する．モデルの形状は HMnet を用い，音素環境に関係なく学習サンプルの音響的特徴の類似性にのみ従って分割をしていく.

4.4.1　環境要因を必要としない音素 HMnet 構成法のアルゴリズム

アルゴリズムの枠組みは逐次状態分割法と同じであり，第 2 章で説明したアルゴリズムの Step 1 と Step 3 を少し修正することで実現される．修正したアルゴリズムを以下に示す．なお，ここでは混乱を避けるために，オリジナルと同様に各状態は 2 混合のガウス分布を持つものとして説明するが，第 3 章で提案した高速化手法を使うこともできる.

Step 1 初期モデルの学習
初期モデルとして，音素毎に 1 状態で出力分布として 2 混合のガウス分布 (対角共分散行列) を持つ HMnet を用意し，すべての学習サンプルを使って学習する.

Step 2 分割すべき状態の決定
すべての状態の中で，出力分布が最も拡がった状態を選ぶ．ここはオリジナルと同様に，2 混合のガウス分布を単一ガウス分布で近似した時の分散の値を計算し，それの最も大きな値を示す状態を分割すべき状態として決定する.

Step 3 状態の分割
選択された状態を 2 つに分割する．この時，新しい状態の出力確率分布は，分割された状態が持っていた 2 混合のガウス分布を 1 つずつ割り当てる．そうしたうえで，新しい状態の配置を時間方向 (直列) に連結した場合の学習サンプルに対する尤度 P_t と，コンテキスト方向 (並列) に連結した場合の尤度 P_c を計算し，より尤度の高い方を採用する．P_t と P_c は，以下のようにして計算される.

- コンテキスト方向への分割
  コンテキスト方向への分割は，パスが 2 つに別れるためにそれぞれの学習サンプルがどちらの状態を通るかを決定する必要がある．ここでは，各学習サンプル 1 つ 1 つについて，尤度の高い方の状態を通るように決定する．

$$P_c = \sum_{y_j \in Y_m} \max(P_m(y_j), P_M(y_j)) \tag{4.1}$$

ここで，

Y_m : 分割すべき状態 m を通る学習サンプルの集合
y_j : 状態 m を通る j番目の学習サンプル
$P_m(y_j)$: y_jを状態 m に割り当てた時の尤度
$P_M(y_j)$: y_jを状態 Mに割り当てた時の尤度

- 時間方向への分割
  ここは，オリジナルの逐次状態分割法と同様に，どちらの状態を前に置くかで 2 とおりの尤度を計算し，より大きい尤度を P_tとして採用する．

Step 4 分布の再推定
オリジナルと同様に，HMnet 全体を再推定し，所定の状態数になるまで Step 2, Step 3 を繰り返す．

Step 5 分布の変更
最終的に使いたい出力分布にするために，HMnet 全体を再学習する．

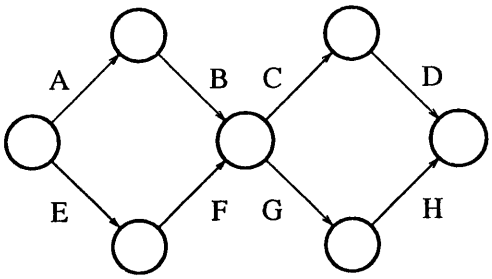
このように修正することで，環境要因を与えずにモデルを構成することができる．オリジナルの逐次状態分割法では，音素環境を指標として，それ毎に分割を行なっていた．それに対し，本手法では各学習サンプルが独立に分割されていく．この時，分割は尤度のみを指標として行なわれるため，各学習サンプルの音響的類似性に基づいた分割であるといえる．こうすることで，すべての環境要因 (音素の環境要因だけではなく) のなかで最も影響のあった要因について分割されることになり，音素の音響的特徴をよく表現したモデルが得られることが期待される．

このアルゴリズムによって得られた HMnet はコンテキスト依存モデルではないが，学習サンプルのコンテキスト情報を活用することでコンテキスト依存モデルのような音素認識もできる．そのためには，HMnet の構造が決定したあとで，“context table” と呼ばれる表 (図 4.2) を作成する．この表はパスの名前と，そのパスが受理する音素環境のリストの対からなる．認識時には，context table と認識対象の音素環境情報からパスを制限し，認識をする．

“context table”(図 4.2) は，HMnet の構造が決定した後で以下のようにして作成する．

1. 各学習サンプルについて，それぞれの音素環境情報を通過するパスに割り当てる
2. 各パス毎に，割り当てられた音素環境をまとめ，context table に書き込む

認識時には，認識対象の音素環境情報を受理するパスを context table から逆引きによってピックアップし，そのパスについてのみ尤度を計算する．この時，音素環境情報によってはパスが複数ピックアップされてくることがあるが，その時はすべてのパスについて尤度を計算し，その中で最大のものを最終的な尤度とする．また，学習サンプルにない音素環境は対応するパスが存在しないために 1 つもピックアップされてこないが，その時は別にコンテキスト非依存 HMM を用意しておき，それの尤度を使うことにする．



context table

path-name	list of contexts
ABCD	a-m+o,a-m+e,...
ABGH	a-m+i, u-m+a,...
EFCD	e-m+o, e-m+e,...
EFGH	u-m+i, a-m+i,...

図 4.2: “context table” の例

4.4.2　環境要因を必要としない音素 HMnet 構成法の特徴

提案手法の大きな利点は，“環境要因を与える必要がない” ことである．コンテキスト依存モデルの構成のように，適切な環境要因を選ばなくても，本方式では自動的に音響的な特徴をよく表現したモデルが生成される．更に 4.3節に述べた理由によりオリジナルの逐次状態分割法では HMnet が構成できない場合でも，提案手法は常に HMnet を構成することができる．また context table を用いることで認識対象音素と同じ音素環境を持つ学習サンプルから推定されたパスのみで認識することができるため，コンテキスト依存モデルと同様に認識対象音素の音素環境情報によって探索空間を絞り込むことが可能である．

一方，オリジナルの逐次状態分割法が持っていた “音素環境の補間作用” という利点は失なわれている．オリジナルの逐次状態分割法では，学習サンプルに現われなかった音素環境に対応するモデルも，ある程度補間作用によって得られる．これは，例えば /a-k+a/ という環境が現れなかった場合，/a-k+i/ の前半部分と，/i-k+a/ の後半部分をとってきて，/a-k+a/ のモデルとする，といった方法で補間していることになる．もちろん，すべての音素環境を補間できるわけではなく，また実際に学習サンプルを集めてきて学習させたモデルよりは性能は落ちると思われるが，十分な学習サンプルが得られなかった時にこの補間作用は有用である．但し，最近では音声資料の収集もすすみ，多量の学習サンプルを用意することが可能になってきた．その結果学習サンプルに現われない音素環境は極少数になり，それらの尤度をコンテキスト非依存の HMM での尤度で代用しても，あまり影響はないと思われる．

4.5　音素認識実験

提案手法の有効性を見るために，提案手法とオリジナルの逐次状態分割法のそれぞれについて，特定話者音素認識実験を行なった．なお，本章ではこれ以降，提案手法を SSS-free，オリジナル

の逐次状態分割法を SSS-original と表記する.

4.5.1　6 子音の認識

まず SSS-free の大まかな性能と話者による差を見るために 6 子音のみの認識実験を行なった. 話者は男性 6 名, 女性 4 名の計 10 名, 認識タスクは/b, d, g, m, n, N/ の 6 子音, その他の条件は 3.4節の実験と同じである. SSS-original には, 環境要因として前後の音素を与えた. また認識時には前後の音素環境は既知であるとし, パスを制限した. HMnet に表現されていない音素環境が現われた時は, コンテキスト非依存 HMM(4 状態, 3 ループ, 単一ガウス分布) での尤度をその音素の尤度とした.

各話者の状態数と認識率の関係を図 4.3から図 4.12に, また状態数が 110 の時の認識率を表 4.1 に示す. SSS-original では 4.3節で述べたように, 与えた要因が不十分である時に分割ができなくなってしまうことがあるが, その時は, その音素についてはそれ以上分割を行なわずに他の音素について分割を行なうようにした. ここで表 4.1中の "*" は, すべての音素が分割できない状況になってしまったためにそこでモデルの構成が止まってしまったことを示す. また, 図 4.3から図 4.12でグラフが途中で切れているものも同様の理由による.

これらを見ると, 10 人の話者を大きく 2 つに分けることができる. 話者 FKN, FKS, MHO, MMY の 4 人は, SSS-original に比べて SSS-free の方が認識率が高い. それに対し, 残りの 6 人は 2 つの手法にそれほど差は見られない. これは, 話者によって音素の変形に関与する要因が違うことを意味していると思われる. つまり, 認識率にあまり差の見られない 6 人は, 音素の変形要因が主に先行音素と後続音素であるために, どちらの手法でもそれら 2 つの要因についての状態

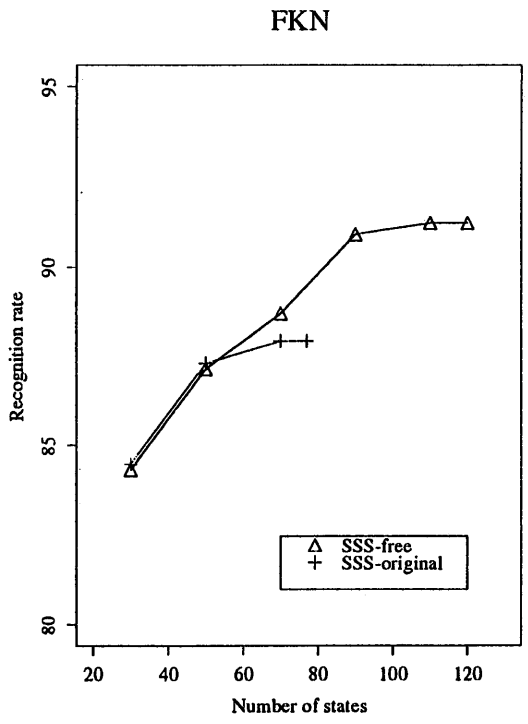


図 4.3: 話者 FKN の 6 子音認識率

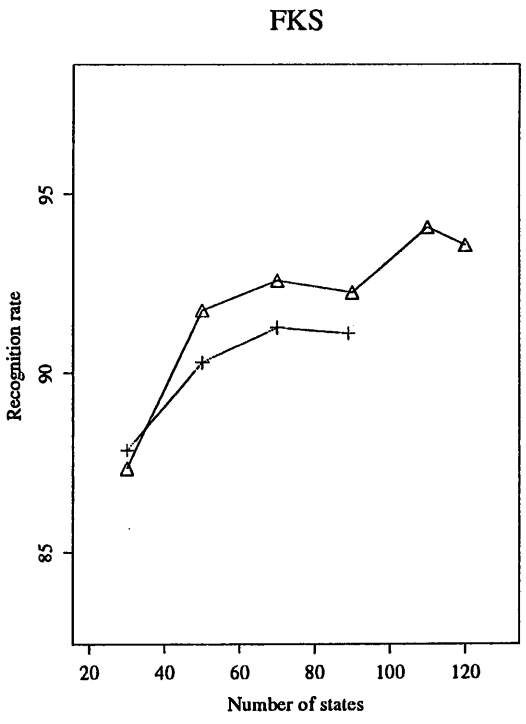


図 4.4: 話者 FKS の 6 子音認識率

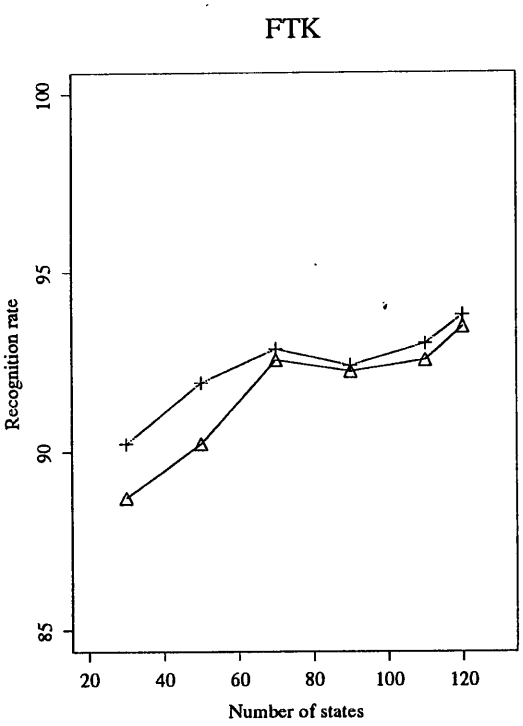


図 4.5: 話者 FTK の 6 子音認識率

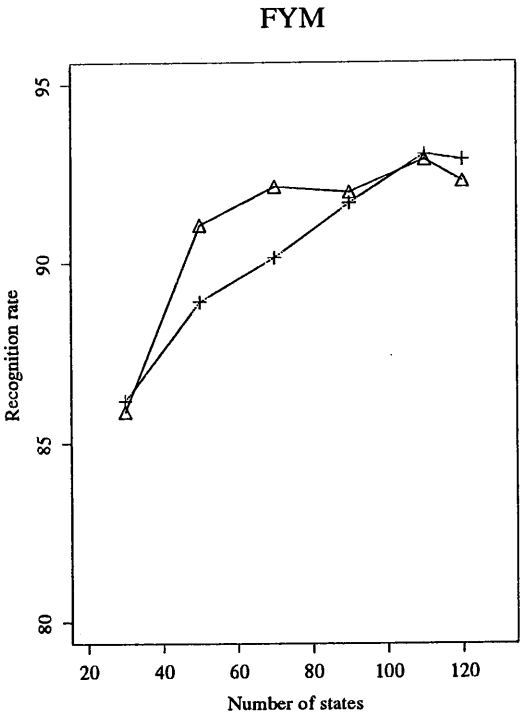


図 4.6: 話者 FYM の 6 子音認識率

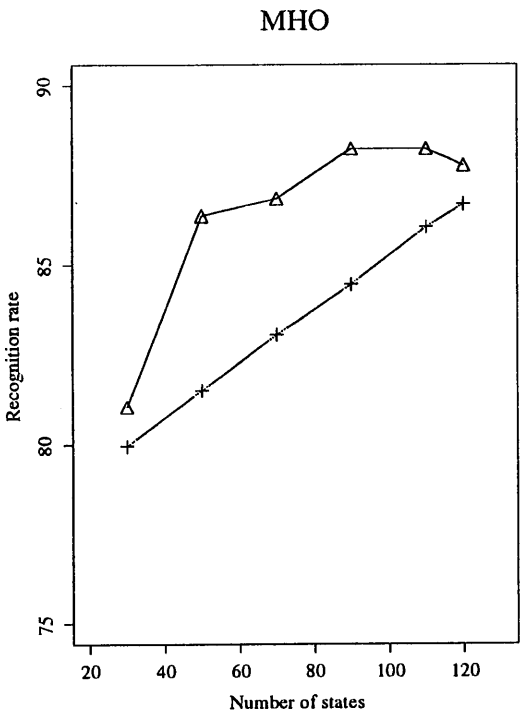


図 4.7: 話者 MHO の 6 子音認識率

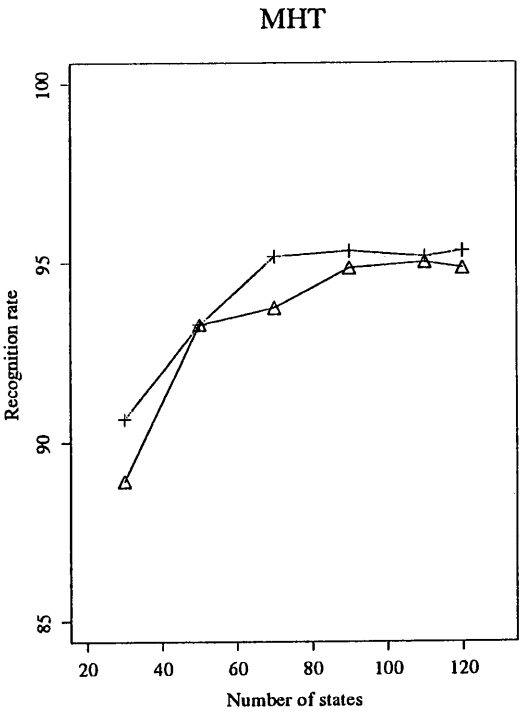


図 4.8: 話者 MHT の 6 子音認識率

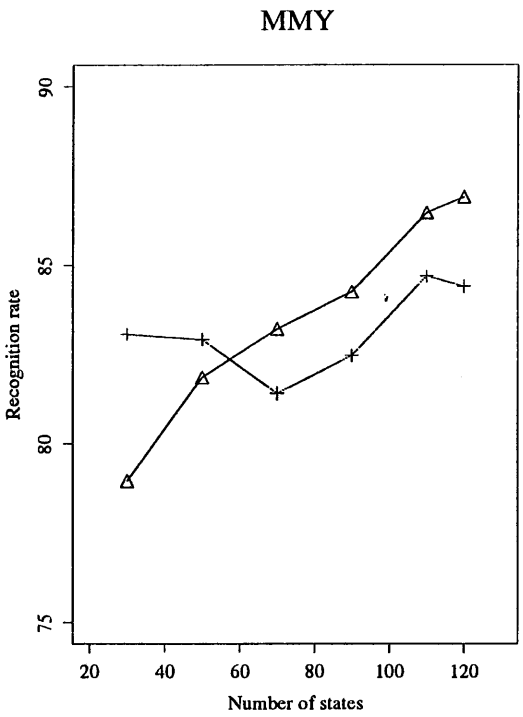


図 4.9: 話者 MMY の 6 子音認識率

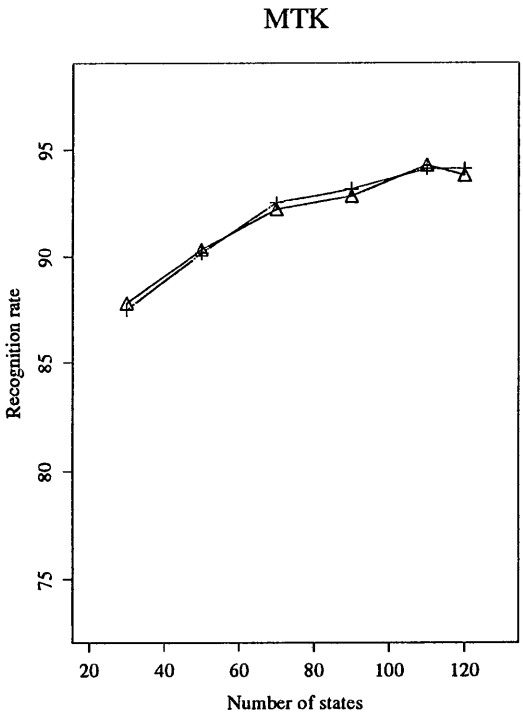


図 4.10: 話者 MTK の 6 子音認識率

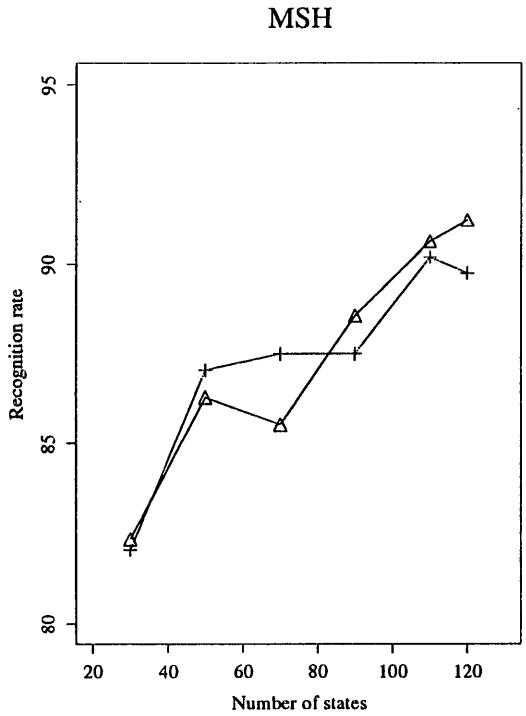


図 4.11: 話者 MSH の 6 子音認識率

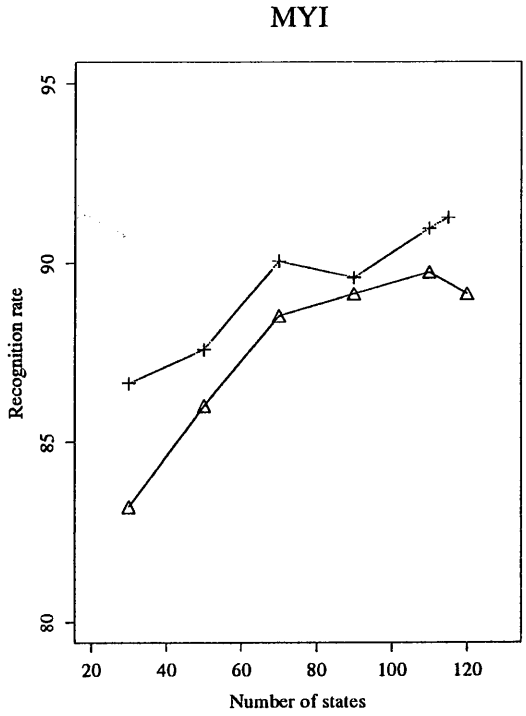


図 4.12: 話者 MYI の 6 子音認識率

表 4.1: 6 子音の音素認識率 (状態数 110)

speaker	SSS-original	SSS-free
FKN	87.9%*	91.2%
FKS	91.1%*	94.1%
FTK	93.0%	92.6%
FYM	93.0%	92.9%
MHO	86.0%	88.2%
MHT	95.2%	95.0%
MMY	84.7%	86.5%
MSH	90.2%	90.6%
MTK	94.1%	94.3%
MYI	91.0%	89.7%
Mean	90.6%	91.5%

分割が行なわれた結果，認識率にあまり差が見られなかった．それに対して認識率に差の見られた 4 人は，音素の変形が先行音素と後続音素以外の要因からの影響もあったと思われる．SSS-free ではすべての要因を考慮した分割が行なわれたため，音響的特徴をよく表現したモデルを得ることができた．一方 SSS-original では先行音素と後続音素しか環境要因として与えていないために無理矢理それらの要因について分割したところ，音響的特徴をよく表現していないモデルになってしまった．その結果，認識率に差が見られたのだと思われる．これは，話者 FKN と FKS では，SSS-original が途中で止まったことからも推察される．

4.5.2　全音素の認識

前節で，話者によって音素の変形に関与する要因が違うことがわかった．本節では音素の変形要因についてよりくわしく見るために，前節で分けられた各グループから 2 人ずつ，計 4 人の話者について全音素での認識実験を行なった．なお，実験を高速に行なうために，第 3 章で提案した高速化を行なった SSS-free，SSS-original で実験した．

認識タスクは /b, d, g, m, n, ɴ, p, t, k, ʧ, ts, s, ʃ, ʒ , z, h, r, w, j, a, i, u, e, o/ の 24 音素，話者は男性 2 名 (MHO, MTK)，女性 2 名 (FKN, FTK) の計 4 名について実験を行なった．各音素を表現する状態数は，最大 30 状態に制限した．その他の条件は前節での実験と同じである．また，前節で述べた，話者 FKN と MHO は環境要因として先行音素と後続音素だけでは不十分である，という仮説の正当性を検証するために，先々行音素と後々続音素をも環境要因として与えたオリジナルの逐次状態分割法[2]についても同様な実験を行なった．

各話者の状態数と認識率の関係を図 4.13から図 4.16に，また状態数が 500 の時の認識率を表 4.2 に示す．これらを見ると，話者 FKN と MHO は，SSS-original の認識率は SSS-free に比べて低くなっている．それに対し，話者 FTK と MTK はどの手法でもあまり差は見られない．これは，前節での結果と同様である．更に，話者 FKN と MHO では環境要因を 5 つに増やす (SSS-org-5) ことで認識率が向上し，SSS-free と同等の性能になっていることがわかる．これは，話者 FKN と

[2]以降，SSS-org-5 と表記する．

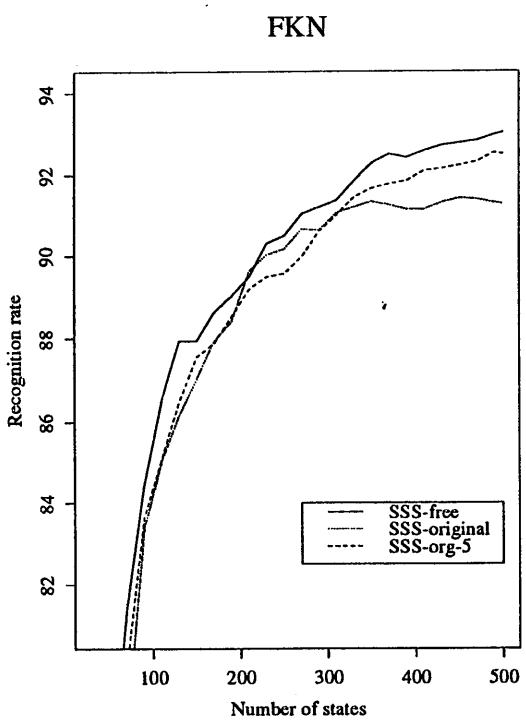


図 4.13: 話者 FKN の全音素認識率

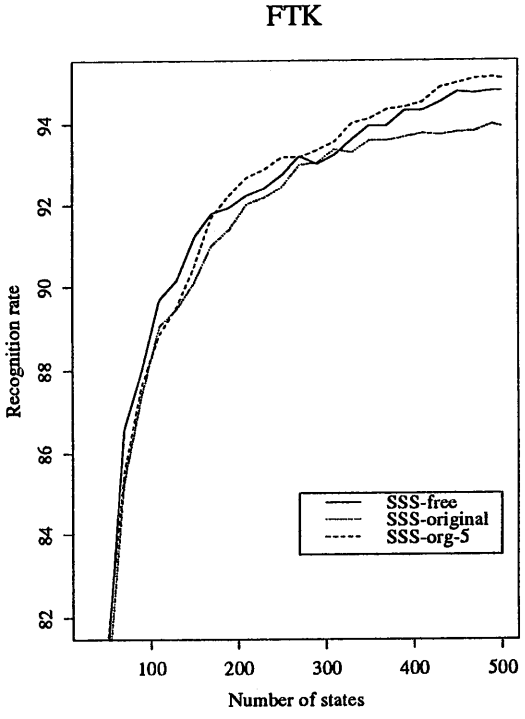


図 4.14: 話者 FTK の全音素認識率

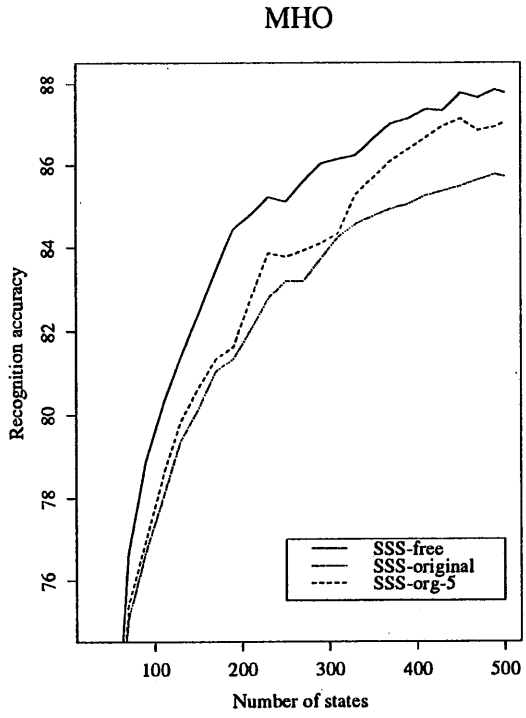


図 4.15: 話者 MHO の全音素認識率

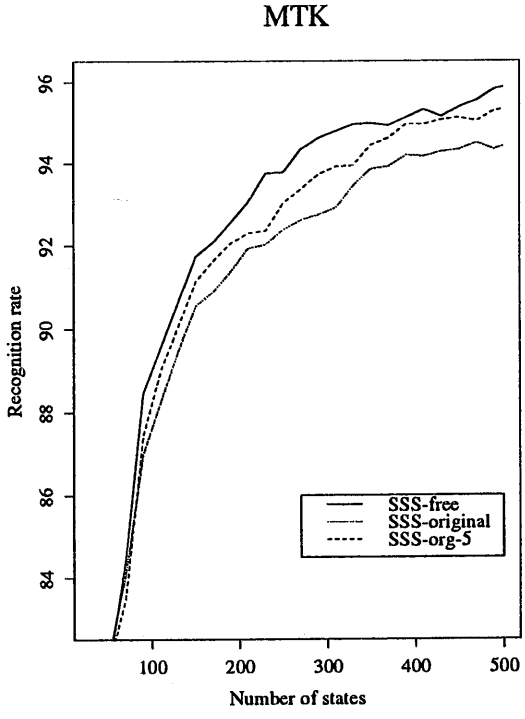


図 4.16: 話者 MTK の全音素認識率

表 4.2: 全音素での音素認識率

speaker	SSS-free	SSS-original	SSS-org-5
FKN	93.9%	91.3%	92.5%
FTK	94.8%	93.9%	95.1%
MHO	87.8%	85.7%	87.0%
MTK	95.9%	94.4%	95.3%
Mean	93.1%	91.3%	92.5%

MHO の発声した音素は先々行音素や後々続音素からの影響も受けているために，SSS-original では表現できなかったような音響的特徴が SSS-org-5 では表現できるようになったためと思われる．また逆に，話者 FTK と MTK では環境要因を増やしても認識率はそれほどかわらず，音素の変形要因が主に先行音素と後続音素のみであることを裏付けている．

4.6 考察

4.6.1 環境要因を必要としない音素 HMnet の構造

SSS-free による HMnet の構造を図 4.17に，また各音素環境がどの状態を通っているかを表 4.3 に示す．これを見ると，/a–b+o/, /e–b+o/ などといった環境は 1 本のパスしか通っておらず，これらの環境は先行音素と後続音素からの影響が主であることがわかる．しかし，/o–b+u/, /u–b+u/ といった環境はすべてのパスを通っており，これらの環境は他の要因からの影響も受けていると思われる．SSS-original では必ず 1 つの環境は 1 本のパスに割り当てられるため，/o–b+u/, /u–b+u/ といった環境の音響的特徴をよく表現することはできない．

また，総じて各音素環境あたり 1 本から数本のパスで表現されており，認識時に認識対象の前後環境情報でパスを制限する方法が有効であろうと考えられる．

4.6.2 音素別認識率

4.5.2節の結果をくわしく見るために，話者 MHO についての音素別認識率を表 4.4に示す[3]．これは状態数 500 の時の結果である．ここで，小括弧内の数字はその音素を表現している状態数，大括弧内の数字はどの環境要因について分割されたかを示しており，[先々行音素，先行音素，後続音素，後々続音素，時間方向] に対応している．また，“*” はその音素の変形要因が与えられていなかったために分割できなくなったことを示している．

表 4.4を見ると，各手法間の認識率の関係から音素を 3 種類に大別することができる．/N/, /h/, /n/, /m/ といった音素は，SSS-free に比べて SSS-original では認識率が悪い．しかし，音素環境を 5 つにすること (5-context) で認識率は向上し，SSS-free と同程度の認識率を示している．これらの音素は主に 5-context からの影響によって変形しているものと思われる．それゆえ，3-context のみを与えた SSS-original では音響的特徴をよく表現したモデルは得られなかったために認識率が落ちている．この結論は，/N/, /h/, /n/ といった音素は 3-context のみ与えた時は分割がで

[3] 環境要因として前後の音素を与えたものを 3-context，先々行と後々続の音素も与えたものを 5-context と表記する．

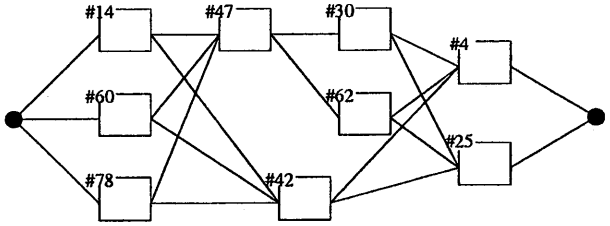


図 4.17: 環境要因を必要としない音素 HMnet(音素 /b/ に対応)

表 4.3: 音素環境が通っている状態 (一部)

phoneme	State No.								
contexts	4	14	25	30	42	47	60	62	78
N–b+u	×	×	○	○	×	○	○	○	○
a–b+i	○	○	×	○	×	○	×	×	○
o–b+u	○	○	○	○	○	○	○	○	○
a–b+o	×	×	○	×	○	×	×	×	○
e–b+a	×	○	○	×	○	○	×	○	○
e–b+e	○	○	×	○	×	○	×	×	×
e–b+i	○	○	×	○	×	○	×	×	×
e–b+o	×	○	○	×	○	×	×	×	×
i–b+a	○	○	○	○	×	○	○	○	○
i–b+e	○	○	×	○	×	○	×	×	○
u–b+u	○	○	○	○	○	○	○	○	○
i–b+i	○	○	×	○	×	○	×	×	○
i–b+o	×	○	○	×	○	○	×	○	○
N–b+a	×	○	○	×	×	○	×	○	○

きなくなっていることからも裏付けされる．次に，/t/, /z/, /i/ といった音素は，3-context だけでなく 5-context を与えた時でも SSS-free より SSS-original の認識率は下がっている．これらの音素は 5-context 以外の要因からの影響で変形を受けているために，5-context を要因として与えても SSS-free に比べるとよいモデルは構成されなかったものと思われる．この結論は，/t/ が 5-context を与えても分割できなくなっていることからも推測される．これらの音素に影響を与えていた要因であるが，3 つ前の音素や 3 つ後の音素は時間的に離れているために影響を強く与えたとは考え難いので，おそらく音素環境以外の要因 (例えば発声速度や発声環境，また語頭，語中の違いなど) が影響していたものと思われる．最後に上に挙げた以外の音素，例えば /b/, /ʧ/, /j/ などは，SSS-original に比べて SSS-free は同程度か若干悪い認識率を示している．これらの音素は 3-context でも 5-context でもあまり差がないことから，音素の変形要因は先行音素と後続音素であると思われる．その結果，どの手法でも同じような HMnet が構成され，同程度の認識性能を持つ．しかし，SSS-original では前後環境情報によって必ずパスが 1 本に制限されるのに対し，

表 4.4: 話者 MHO の音素別認識率

Ph.	SSS-free	SSS-original	
		3-context	5-context
b	68.52% (12)	72.22% (15) [0,6,5,0,3]	68.52% (13) [0,4,4,1,3]
d	77.66% (20)	86.17% (26) [0,13,8,0,4]	85.11% (25) [4,7,4,5,4]
g	58.88% (20)	63.55% (21) [0,9,6,0,5]	59.81% (18) [2,3,4,5,3]
m	88.89% (19)	85.19% (22) [0,7,9,0,5]	88.15% (20) [5,1,6,3,4]
n	91.54% (24)	85.07% (28)* [0,11,9,0,7]	91.54% (28) [6,5,3,7,6]
N	94.44% (25)	90.74% (19)* [0,1,10,0,7]	94.44% (27) [1,1,17,1,6]
p	33.33% (4)	50.00% (4) [0,1,0,0,2]	50.00% (4) [0,0,1,1,1]
t	95.14% (30)	86.11% (15)* [0,7,4,0,3]	88.19% (15)* [3,2,2,4,3]
k	92.75% (30)	93.78% (30) [0,9,13,0,7]	90.16% (24) [2,4,7,4,6]
tʃ	57.14% (13)	75.00% (14) [0,2,5,0,6]	71.43% (13) [0,1,3,2,6]
ts	73.68% (10)	73.68% (10) [0,2,0,0,7]	68.42% (9) [0,1,0,1,6]
s	96.39% (29)	97.59% (29) [0,9,8,0,11]	96.39% (29) [3,5,6,4,10]
ʃ	97.92% (23)	97.92% (26) [0,7,8,0,10]	97.92% (24) [2,4,5,2,10]
ʒ	87.88% (20)	84.85% (21) [0,9,4,0,7]	90.91% (21) [3,4,4,2,7]
z	77.27% (9)	68.18% (12) [0,5,1,0,5]	72.73% (10)* [1,2,0,2,4]
h	85.42% (21)	50.00% (11)* [0,3,3,0,4]	85.42% (27) [3,2,4,6,11]
r	91.00% (27)	92.89% (30) [0,15,12,0,2]	94.31% (27) [3,5,8,8,2]
w	61.11% (2)	83.33% (2) [0,0,0,0,1]	61.11% (2) [0,0,0,0,1]
j	90.91% (12)	95.45% (15) [0,5,4,0,5]	95.45% (14) [3,1,5,1,3]
a	91.74% (30)	89.85% (30) [0,9,14,0,6]	89.67% (30) [0,7,12,4,6]
i	88.56% (30)	84.84% (30) [0,7,16,0,6]	82.98% (30) [2,4,15,1,7]
u	79.92% (30)	80.33% (30) [0,7,16,0,6]	81.97% (30) [1,5,14,2,7]
e	89.56% (30)	87.54% (30) [0,7,15,0,7]	88.55% (30) [2,5,11,4,7]
o	90.36% (30)	88.98% (30) [0,10,15,0,4]	90.63% (30) [0,8,14,2,5]

表 4.5: 尤度計算を行なった平均パス数

#states	MHO		MTK	
150	34.8	7.43%	34.9	9.34%
250	46.2	4.67%	47.3	3.34%
350	59.5	3.97%	57.3	2.70%
450	70.5	3.30%	68.7	2.55%
500	75.2	3.04%	73.9	2.47%

SSS-free ではパスが 1 本になるとは限らない．そのため，SSS-free では探索空間が SSS-original に比べて拡がってしまい，多少の認識率低下を招いたものと思われる．

このようにしてみると，同一話者の発声した音素でも音素の種類によって変形に関与する環境要因には差があることがわかった．また，音素によっては先々行，後々続音素まで含めた音素環境要因以外の要因からの影響があることもわかった．

4.6.3　認識時の計算時間

SSS-original は認識対象音素の前後環境情報を与えることで，常にパスを 1 本に制限することができる．しかし，SSS-free では必ずしも 1 本に制限されないので，パスの数だけ尤度を計算する必要があり，計算時間が SSS-original に比べて遅くなってしまう．ここでは，実際に得られた HMnet で，パスがどの程度制限されるかを見る．

話者 MHO と MTK について，認識時に尤度計算を行なったパス数の平均値と，HMnet 全体に占める割合を表 4.5に示す．SSS-original では必ず音素あたり 1 本に制限されるので，表中のパス数を 24 で割った値が SSS-original に比べての計算時間になる．これを見ると，状態数が増えるに従って尤度を計算すべきパス数は増加しているが，HMnet 全体に占める割合は確実に減っており，前後環境情報によるパスの制限が有効であることがわかる．また，SSS-original に比べた時の計算時間は，状態数 500 の時でおおよそ 3 倍程度でおさえられている．

4.6.4　学習サンプルの質による頑健性

学習サンプルデータは，ノイズのない場所で綺麗な発声によって得られるものを使うのが理想であるが，なんらかの原因でノイズによる変形を受けたような学習サンプルが混入する可能性がある．また，現在音声データのラベル付け (音素区切りの決定) は人手によってされているが，その過程でのラベルミスや判断のゆらぎは避けられない．そうした “汚れた” 学習サンプルが混入していた場合のモデルの頑健性について考察する．

SSS-original では，同じ音素環境を持つサンプルは必ず同じパスを通る．そこで，同じ音素環境を持つ “汚れた” サンプルと “綺麗な” サンプルはどこまで分割しても同じパスに割り当てられる．その結果，“綺麗な” サンプルだけで学習した時と HMnet の形状はほぼ同じになると思われるが，“汚れた” サンプルが混入している分，出力確率密度関数の分散が拡がったものになり，モデルの精度は落ちてしまう．

それに対し，SSS-free では音素環境に関係なくサンプルの音響的特徴に従って状態分割が行なわれていくために，“汚れた” サンプルと “綺麗な” サンプルは違うパスに割り当てられることに

なる．その結果，“綺麗な” サンプルだけで学習した HMnet とは違う構造を持つ HMnet が得られることが予測される．しかし認識時にはすべてのパスの中で最も尤度の高いものが選ばれるので，“綺麗な” サンプルが割り当てられていたパスの尤度が採用されることが期待される．つまり，“汚れた” サンプルの認識性能に対する影響は，SSS-original に比べると少ないと思われる．

4.7　まとめ

本章では，コンテキスト依存モデルの本質的問題点である，“考慮すべき環境要因を前もって与える” という点を解決するために，環境要因を必要としない HMnet の構成法を提案した．音素の認識実験によると，話者や音素によって音素の変形に関与する要因は違う．そして，従来最も多く用いられてきた先行音素と後続音素という環境要因だけでは不十分な場合が多いことがわかった．コンテキスト依存モデルを構成する時には，適切な環境要因を調査し，それを過不足なく与える必要がある．そうしないと，モデルが音素の音響的特徴をうまく表現できず，認識性能が低下してしまう．特にオリジナルの逐次状態分割法では，もし与えた環境要因が不十分であると，分割できなくなったり，音響的な特徴をよく表現していないモデルになったりしてしまう．ところが提案手法では，音素変形に関与する要因を与える必要がないので，どのような音声であっても常に音響的特徴をよく表現したモデルを自動的に構成することができる．平均の誤認識率は，オリジナルの逐次状態分割法による HMnet での誤認識率に比べて 21%程度減少した．また認識時の計算時間は，オリジナルの逐次状態分割法に対して状態数 500 で 3 倍程度でおさえられることがわかった．

第5章

離散型 HMnet を用いた言語モデル

5.1 はじめに

現在の音声認識システムは，音素認識率が特定話者で85%から90%程度であり，なんらかの言語的制約を導入しないと実用に耐えうる認識率を達成することはできない．そこで本章では，よりよい言語モデルを開発することを目的とする．

従来よく使われている言語モデルには，文法的な知識を与えるモデルと，統計的なモデルの2つがある．前者の代表的な例としては，有限状態オートマトンや文脈自由文法などがある．これらは，与える文法がよくタスクを表しているならばよいモデルとなる．特にCFGは拡張LRパーザと組み合わせることで，直接認識部を駆動する方法 [1] が提案されており，よい言語モデルとして注目されている．しかし，これらの文法的モデルでは，文法を人が手で与える必要があるために，構成するのに大変な労力と時間がかかるのが大きな問題である．これに対し，統計的なモデルは多量のサンプルから自動学習できるために構成は容易であるが，よい文法を与えられたモデルに比べると絞り込み能力が弱く，また学習サンプル数に対して推定すべきパラメータ数が多い場合には，学習サンプルに依存したモデルが構成されるという欠点を持つ．

本章では2つのアプローチのうち“自動学習できる”という利点に注目し，より強い絞り込み能力を持つ統計的言語モデルを提案する．

5.2 従来の統計的言語モデル

従来からよく用いられている統計的言語モデルには，次の2つがある．

- n-gram モデル
- ergodic HMM

各モデルの処理単位としては音韻や単語，文節など様々なものが提案されているが，ここでは処理単位を単語として各モデルを説明する．

n-gram モデルは対象言語を $n-1$ 重のマルコフ過程であるとみなし，$n-1$ 個の単語が観測された時に，次に続く単語の条件付き確率によって言語を表現するモデルである．このモデルの学習は，多量の学習サンプル中から数え上げによって条件付確率を推定するために非常に容易であ

り，また追加学習も簡単にできる．しかし，n を増やすと推定すべき条件付確率は指数関数的に増加するため，通常は n として 2(bigram と呼ぶ) や 3(trigram と呼ぶ) が選ばれる．このモデルは学習の簡単さからよく使われるモデルであるが，現実には n を大きくとれないために遠くの距離にある単語間の相関を表現することはできず，また文中のどの位置によく出現する単語かといった時間的な記述能力にも欠ける，といった問題点がある．

ergodic HMM とは，各状態間の遷移をすべて許したような HMM であり，遷移するたびに単語を 1 つ出力する．このモデルも状態を増やすとパラメータの推定が大変になるため，計算量の問題からあまり大規模なモデルは構成できない．そこで，状態数を逐次増加させていくことで計算量を削減した学習アルゴリズムも提案され [9]，bigram を越える性能が得られている．このモデルも n-gram モデルと同様に，モデルが小規模な時は遠くの単語間の相関を表現することはできず，また ergodic な結合をしているので，時間的な記述能力に欠ける．

従来の言語モデルには，以上に述べたような問題点がある．そこで本章ではこれらを解決するために HMnet を言語モデルとして用いることを考える．HMnet は状態遷移の方向が 1 方向であるため，各状態が文中での位置と対応することになり，時間的な記述能力を持つ．また，パスが並列に複数ある構造のため，遠い位置にある単語間の相関も表現することが可能である．

5.3　離散型逐次状態分割法

オリジナルの逐次状態分割法は出力分布として連続ガウス分布を持っていた．いま HMnet を言語モデルとして適用する時，出力としては単語を発生するために離散分布を持つ必要がある．そこで，逐次状態分割法を改良し，離散分布を持つ HMnet を構成するようにする．

離散型逐次状態分割法のアルゴリズムを以下に示す (図 5.1)．なお学習サンプルは単語系列であるとする．

Step 1 初期モデルの学習
初期モデルとして，1 状態で離散分布を出力分布として持つ HMM を用意し，すべての学習サンプルを使って学習する．

Step 2 分割すべき状態の決定
すべての状態の中で最も評価値 V_i の大きい状態を分割すべき状態として決定する．

Step 3 状態の分割
Step 2 で決定された状態を 2 つに分割する．この時，新しい状態の出力確率分布は分割前の出力分布に小さな乱数を加えたものとする．また新しい状態を直列に接続するか並列に接続するかで 2 とおりの可能性があるが，それぞれの可能性について実際に学習サンプルを使って HMnet を学習し，その時の尤度が大きい方を採用する．

- 時間方向への分割
  この時は学習サンプルを振り分ける必要がないため，状態を直列に接続した HMnet を用意し，すべての学習サンプルを使って HMnet 全体を再学習する．
- コンテキスト方向への分割
  この時はパスが 2 つに別れるため，分割前の状態を通っていたすべての学習サンプルをどちらかのパスに振り分ける必要がある．これは以下のようにして行なう．

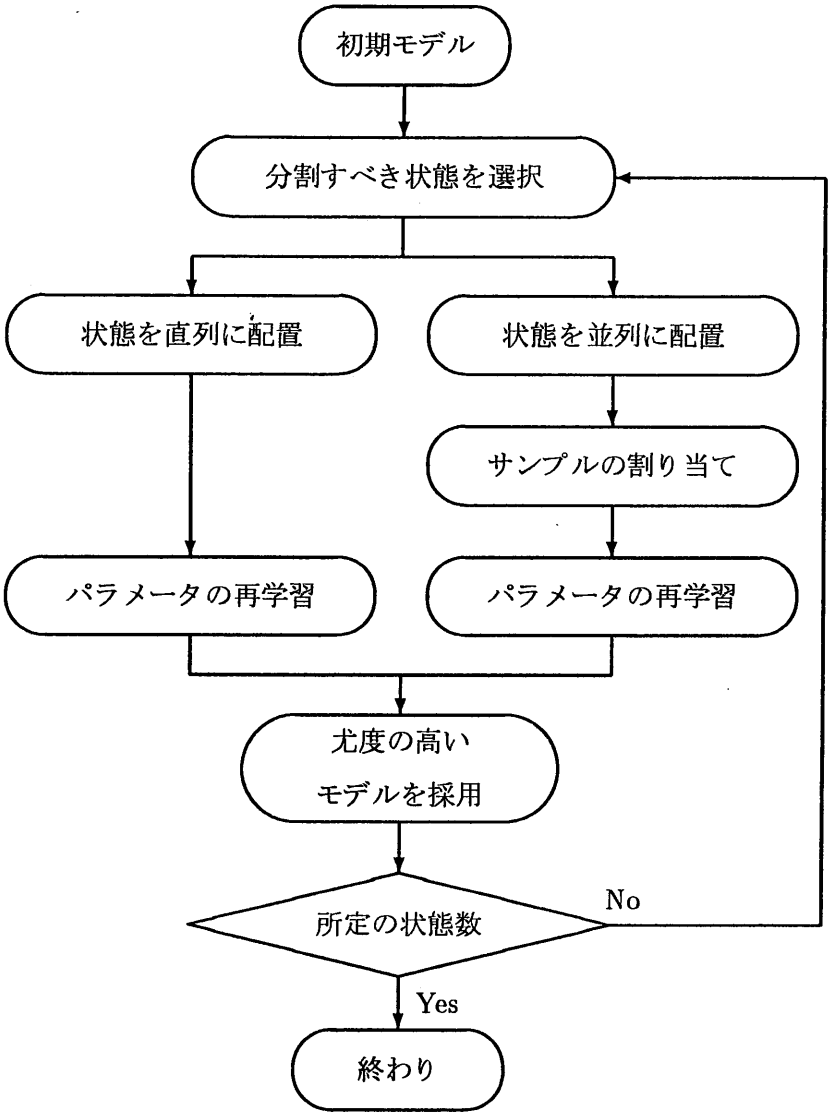


図 5.1: 離散型逐次状態分割法のアルゴリズム

1. 分割すべき状態を通っていたすべての学習サンプルについて，Viterbi アルゴリズムによってその状態が出力する部分系列を切り出してくる．
2. 各部分系列を 2 つにクラスタリングする．

こうして学習サンプルをどちらかのパスに割り当てたあとで，すべての学習サンプルを使って HMnet 全体を再学習する．

Step 4 HMnet の選択

Step 3 で計算された 2 つの HMnet のうち，学習サンプルに対する尤度の高い方を採用する．所定の状態数になるまで，Step 2, Step 3 を繰り返す．

このようにして，離散型の HMnet を構成することができる．

このアルゴリズムでは，

1. 分割する状態を決定する際の評価値 V_i

2. 状態分割の際のクラスタリング手法

3. クラスタリングの際のサンプル間の距離尺度

の 3 点を決定しなければならない．これらについては，5.4節の実験で各種手法について HMnet を構成し，perplexity で評価する．

5.4　離散型 HMnet の性能評価

離散型 HMnet の言語モデルとしての性能を評価するために，言語モデルの構成実験を行なった．

5.4.1　実験した手法

解決すべき 3 つの問題点は，以下の手法について実験した．

1. 分割する状態を決定する際の評価値 V_i
   基本的には離散出力分布 $P_i(j)$ $(1 \leq j \leq N, N$は単語数$)$ の形状が拡がったものに高い評価値を与える．また，統計的な頑健性をはかるために，その状態を推定するのに使った学習サンプル数 n_iも考慮する．評価値として以下の 2 つを考える．

   - 出力分布のエントロピー　　(a)
     出力分布の拡がりをエントロピーを尺度として評価する．

$$V_i = n_i \times \sum_j^N \{-P_i(j) \log P_i(j)\} \tag{5.1}$$

   - 単語間の距離を考慮した歪　　(b)
     ある 2 つの単語が文中で同じ使われ方をされることがある[1]．この時，この 2 つの単語は 1 つの状態で表現されるのが望ましいが，上記のエントロピー尺度では単語間の類似性を考慮にいれていないために，そのような状態にも高い評価値を与えてしまい，結果として 1 つの状態で表現すべき単語群を分割してしまう．そこで，文中の使われ方を表現するような単語間の距離を定義し [15]，それを考慮して分割すべき状態を決定する．単語間の距離 $Wd(i,j)$ を次のように定義する．

$$Wd(i,j) = K(Pp_i, Pp_j) + K(Pn_i, Pn_j) \tag{5.2}$$

     ここで，Pp_iは，単語 i が観測された時の，1 つ前に観測された単語の条件付確率分布，Pn_iは，単語 i が観測された時の，1 つ後に観測された単語の条件付確率分布である．また，$K(P,Q)$ は 2 つの離散確率分布 P,Q 間の Kullback Divergence と呼ばれる距離を計算する関数である．

$$K(P,Q) = \sum_j^N \{P(j) \log \frac{P(j)}{Q(j)} + Q(j) \log \frac{Q(j)}{P(j)}\} \tag{5.3}$$

[1] 数字や同じ概念を表す名詞などは，典型的な例である．

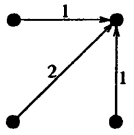


図 5.2: DP パスにかかる重み

このようにして単語間の距離を定義し，分割すべき状態の評価値として以下の式を用いる．

$$V_i = n_i \times \min_j \sum_k^N \{Wd(j,k) \times P_i(k)\} \tag{5.4}$$

2. 状態分割の際のクラスタリング手法
今クラスタリングの対象とする要素は単語系列なので，複数の要素の平均量を定義できない．また，状態が 1 つ増えるたびにクラスタリングをする必要があるため，計算量の多いクラスタリング手法は使えない．そこで，従来から各種提案されているクラスタリング手法の中から最大距離クラスタリング法を採用する．このクラスタリング手法は，まず全サンプル中で互いに最も離れたサンプルの組を 1 組探し出し，それをクラスタ中心として残りのサンプルをクラスタリングする．ただし，孤立点を拾わないために，1 つのクラスタに含まれるサンプル数が閾値以下の場合は，そのクラスタに属するサンプルを除いてもう一度クラスタ中心を決定しなおす．本実験では，閾値を 3 に設定した．

3. クラスタリングの際のサンプル間の距離尺度
サンプル間の距離尺度として以下の 2 つを考える．

- Kullback Divergence ..... (α)
サンプルの部分系列中の単語を数え上げ，離散分布にする．その分布間の距離を式(5.3)に従って計算する．この距離は部分系列内の時間的遷移を考慮していない距離である．
- DP 距離 ..... (β)
式 (5.2)で定義される単語間の距離 $Wd_{i,j}$を使って，サンプル間の DP 距離を求める．ここでは傾斜制限なしの対称型 DP 距離 (図 5.2) を用いる．
2 つのサンプル$\boldsymbol{l} = l_1, l_2 \cdots l_L, \boldsymbol{m} = m_1, m_2 \cdots m_M$ の傾斜制限なしの対称型 DP 距離 $D(\boldsymbol{l}, \boldsymbol{m})$ は以下の漸化式で定義される．

$$d(1,1) = Wd(l_1, m_1) \tag{5.5}$$

$$d(i,j) = \min \begin{cases} d(i-1,j) + Wd(l_i, m_j) \\ d(i,j-1) + Wd(l_i, m_j) \\ d(i-1,j-1) + 2 \times Wd(l_i, m_j) \end{cases} \tag{5.6}$$

$$D(\boldsymbol{l}, \boldsymbol{m}) = \frac{d(L,M)}{L+M-1} \tag{5.7}$$

本章では，上記の手法のうち，単語間距離を使わないもの ((a),(α):方法 1) と，単語間距離を使うもの ((b),(β):方法 2) の 2 つの組み合わせについて実験を行なう (表 5.1).

表 5.1: 実験した手法

決定すべき問題	方法 1	方法 2
状態分割の評価値 V_i	出力分布のエントロピー	単語間距離を考慮した歪
クラスタリング手法	最大距離クラスタリング	
距離尺度	Kullback Divergence	DP 距離

5.4.2　言語モデルの性能評価値

言語モデルの性能を評価する値として，ここでは perplexity を用いる．perplexity とは情報理論的な意味での平均分岐数を表しており，ある言語 L の 1 単語あたりのエントロピーを $H(L)$ と表すと，言語 L の perplexity$F_p(L)$ は式(5.8)によって計算される．

$$F_p(L) = 2^{H(L)} \tag{5.8}$$

perplexity$F_p(L)$ の値は，言語 L では，次の単語を $F_p(L)$ 個の等出現確率の単語から決定する必要があることを示している．例えば，n 種類の単語がランダムに連なる任意の長さの系列からなる言語の perplexity は n である．また，ある言語モデルのエントロピーは，McMillan の定理によると，その言語モデルが発生した十分長い具体的な系列$\boldsymbol{x} = x_1, x_2 \cdots x_n$を使って，式(5.9)のように計算できる．

$$H(L) = \frac{-\log p(\boldsymbol{x})}{n} \tag{5.9}$$

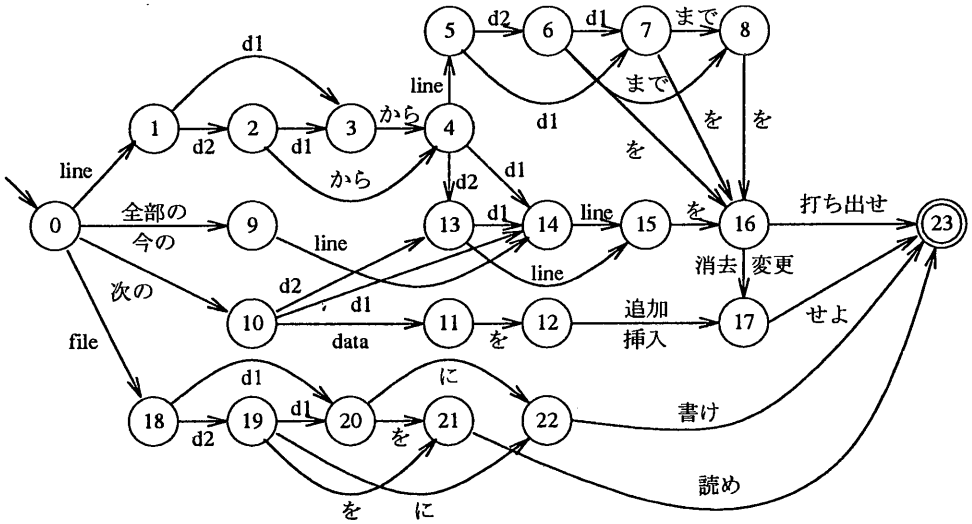
そこで，perplexity を言語モデルの性能を示す尺度として用いる．モデルを構成するためのサンプルとは別に，言語 L からのサンプル (test set) を用意し，構成した言語モデルが test set を生成する確率から式(5.8)，式(5.9) によって perplexity を求める．この値が言語 L を表す真のモデルの perplexity に近ければ近いほどよいモデルであるといえが，一般に真のモデルは未知であるので真のモデルの perplexity も計算できない．このため perplexity は，各モデル間の良し悪しを比較するための相対的な尺度として用いられる．

5.4.3　HMnet の構成実験

HMnet の性能を簡単に評価するために，比較的簡単な言語を対象に HMnet の構成実験を行なった．対象言語は図 5.3の確率有限状態オートマトンで表現されるエディタ制御コマンド (単語数 36) である．また，比較のため同じ統計的言語モデルとしてよく用いられている n-gram モデルについても実験した．

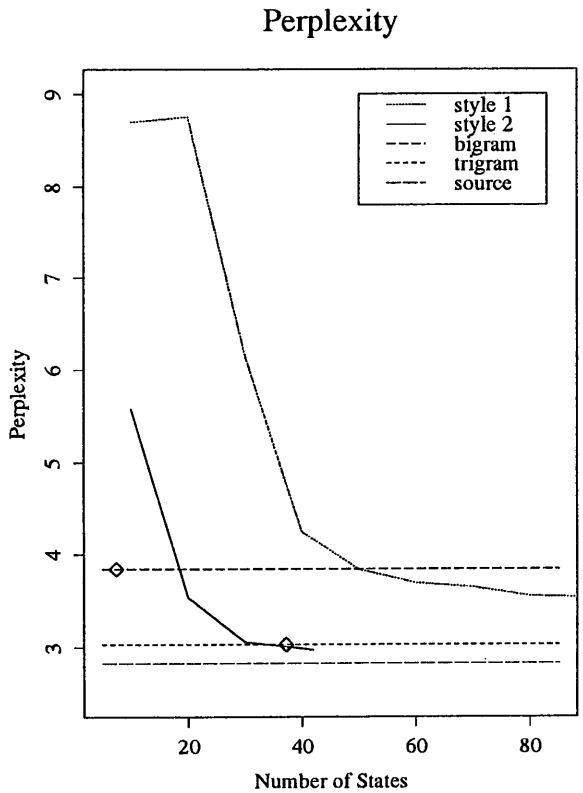
図 5.4に各モデルの perplexity を示す．これを見ると，方法 1 に比べて方法 2 の方が圧倒的によいモデルを構成していることがわかる．これは状態分割の際に DP 距離を用いているために，サンプル間の距離に時間的な関係も反映され，よりよい状態分割が行なわれたことによると思われる．また，方法 2 が状態数 42 で終了しているのは，分割すべき状態を通るサンプルがすべて距離 0 になったからである．つまり状態数 42 程度で対象言語をほとんど表現できたと思われる．

そこで，方法 2 で構成した HMnet の構造がどのようになっているかを調べた．42 状態での HMnet を図 5.5に示す．ここで，出力する単語の分布は出力確率の高いもののみを表示しているので，出力確率の和が 1 にならない状態もある．また，#5, #35, #43 は，状態番号の隣に表示



ここで，d1 は一の位の数字を，d2 は十の位の数字を表す．
また，各状態の遷移確率はすべて等確率である．

図 5.3: エディタ制御コマンドの文法



bigram,trigram は，◇の位置における状態数の HMnet と同数のパラメータを持つ．

図 5.4: perplexity の比較

してある確率で自己ループの遷移を持つ．それ以外の状態は確率 1 で隣りの状態へと遷移していく．これを見ると，ほぼ真の文法を表現していることがわかる．また単語間の距離を考慮した分割のため，例えば #2 では,「今の」と「全部の」という単語が表現されており，これは真の文法での状態 0 から状態 9 への遷移を忠実に表現している．また，#23 と#41 は本来 1 つの状態で表現すべきものであるが，これが分離してしまったのは，#41 で表現されている単語がでてくる学習サンプル中に，本来出現する可能性のある単語列がたまたま含まれておらず，比較的早い段階で，それの違いによって他のサンプルと分離してしまったためと思われる．また，#5 や#35 に自己ループの遷移が残っている原因も，本来別のパスで表現されるべき学習サンプルが早い段階で極少数まぎれ込んだためだと思われる．

このように逐次状態分割法は top-down 的な手法であるために，一度分離されてしまうとけして統合されることはなく，無駄なパスを生成してしまう．この問題を解決するために，状態の分割だけでなく融合もしながら HMnet を構成する方法 [16] が提案されており，離散型 HMnet の構成にもその手法をとりいれるか検討する必要があると思われる．

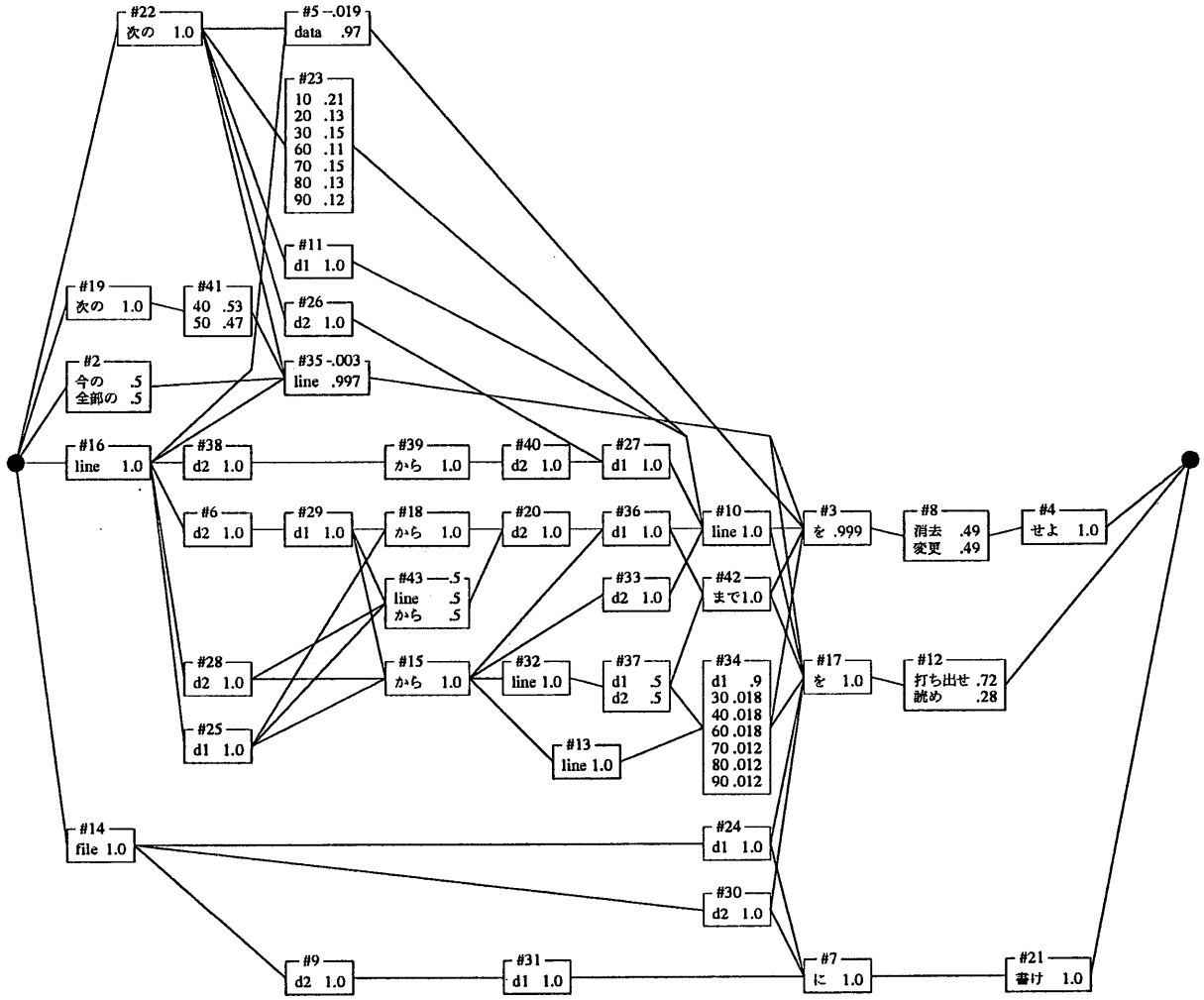


図 5.5: 方法 2 で構成した HMnet

5.5　まとめ

HMnet を言語モデルとして適用するために，離散型の逐次状態分割法を提案した．状態数 24 の有限状態オートマトンが受理する言語 (単語数 36) を対象に言語モデルを構成したところ，パラメータ数が同程度で trigram モデルを越える性能を有する HMnet を得ることができ，また得られた HMnet の構造も，真の文法構造をよく反映していた．

今後は，アルゴリズムの問題点として今回実験した手法が妥当かどうかの検討や，自然言語へ適用した時の能力の評価をする予定である．

第 6 章

結論

6.1 本研究の成果

自然な発話を高精度に認識する連続音声認識技術は，様々な機器が氾濫している現代において重要な技術であり，さかんに研究が行なわれている．本論文では，より高精度な連続音声認識システムを構築することを目的とし，音素認識部と言語処理部のそれぞれについて高精度化をはかった．

HMnet は HMM を基本としたモデルであり，従来よく用いられてきた left-to-right 型の HMM と，ergodic HMM の中間的な構造を持つために，両者の利点をあわせもっている．そこで，音素モデルとして，また言語モデルとして HMnet を用いることで，高精度な連続音声認識を実現した．本研究の主な成果を以下にまとめる．

(1) 逐次状態分割法の高速化

逐次状態分割法は HMnet を効率的に構成するアルゴリズムである．この方法は HMnet の構造をすべて尤度を基準にして自動的に決定することができる．しかし，アルゴリズムの性質上，通常の HMM を学習するのに比べると膨大な学習時間が必要になる．

逐次状態分割法の学習に時間がかかる原因は，

1. 状態が 1 つ増えるたびに HMnet 全体を再学習する
2. 出力分布として混合数 2 のガウス分布を持つ

という 2 点である．そこでアルゴリズムの本質を変えない方向での高速化をはかるために，出力分布として単一ガウス分布を持つ逐次状態分割法を提案した．その結果，認識率は同程度のまま，計算時間を状態数 80 の時で 1/5 程度にまで削減することができた．この実験は 6 子音についてのみの実験であったが，学習サンプルの多い母音を含むような HMnet の構成，また状態数を多くとるような HMnet の構成には，より効果が現われるものと思われる．

(2) 環境要因を必要としない音素 HMnet 構成法

音素の音響的特徴は発声された環境 (前後の音素や発声速度など) によって変形されやすいので，音素の認識には音素環境毎に構成されたモデルがよく用いられる．しかし，同じ音素環境を持つ

音素は本当に似た音響的特徴を有するかどうかは確認されていない．したがって，もし同じ音素環境を持つ音素でも音響的特徴の違うものが含まれていると，高精度なモデルは期待できない．

そこで，音素環境を与えずに，学習サンプルの音響的特徴に基づいて HMnet を構成するアルゴリズムを提案した．その結果，オリジナルの逐次状態分割法による HMnet よりも音響的特徴をよく表現した HMnet が得られ，平均の誤認識率はオリジナルの逐次状態分割法で構成した HMnet での誤認識率に比べて約 21%低減させることができた．また，音素を変形させる要因は話者や音素によって違うことがわかった．

(3) 離散型 HMnet による言語モデルの構成法

従来の統計的言語モデルである n-gram モデルや ergodic HMM は，現在の計算機で実現できる程度の規模では時間的な記述能力に乏しい．そこでこれらのモデルに比べて時間的な記述能力を持つ HMnet を言語モデルに適用するため，離散型の HMnet を構成する逐次状態分割法を提案した．

状態数 24 の確率有限状態オートマトンが生成する文 (単語数 36) をタスクとした実験で，trigram モデルを越える性能があることがわかった．

6.2　今後の課題

本論文では，連続音声の高精度な認識を目的とし，音素認識部，言語処理部のそれぞれについて HMnet を用いた手法を提案し，その有効性を確認した．ここでは最後に，自然な発声を認識するために必要とされる課題を上げておく．

- 音素認識部と言語処理部の融合
  本論文では，音素認識部と言語処理部を別々に開発した．しかしこれらの処理は，逐次的に行なうよりも，一体化した処理として行なう方がよいと思われる．それは，各処理部間で通信する必要がないために音素ラティスなどの中間コードが必要がなく，また言語的な制約から認識部を駆動することができるなど，相補的な認識をすることができるからである．実際，CFG を使って音素認識部を駆動する方法が提案されており [1]，よい認識率を示すことが報告されている．そこで，本論文で開発した手法を融合するアルゴリズムを開発する必要がある．

- 話者適応
  同じ音素を発声しても，話者によって音響的特徴がかなり変わることが知られている．そこで，高精度な認識を行なうために発声者の声の質にモデルを動的に適応する必要があり，様々な研究がなされている．しかし，第 4 章の結論から話者によって音素を変形させる要因が違うため，HMnet の構造自体が違うことがわかっている．従来からやられている話者適応の方式はいずれもモデルの構造は話者に共通としているため，よい話者適応方式とはいい難い．そこで，話者による構造の差をうまく表現できるような話者適応方式を開発する必要がある．

- 意味的処理と対話の管理
  本論文で提案した言語モデルは，あくまでも単語の並びを記述しているだけであり，意味的解釈はしていない．しかし，音声認識を人と機械のインターフェースとして考えた時，人が

話した内容を機械が理解する必要があり，また人との対話を実現する必要がある．そこで，認識結果から意味的情報を取り出し，対話を管理する処理部を開発する必要がある．

以上のことが，人と機械との自然な対話を実現するための残された課題である．

謝辞

本研究を進めるにあたり，数多くの御指導とともにこの研究の機会を与えて下さりました東北大学工学部通信工学科阿曽弘具教授に心から感謝致します．また音声認識の分野におきましては，東北大学情報科学研究科牧野正三助教授，北陸先端科学技術大学院大学下平博助教授，東北大学情報処理教育センター伊藤彰則助手，北陸先端科学技術大学院大学中井満氏，日本電信電話株式会社粟津辰功氏に終始御指導戴きましたことを深く感謝致します．

大学院の音声ゼミにおきましては，東北大学電気通信研究所曽根敏夫教授，同鈴木陽一助教授，東北大学工学部金井浩助教授に貴重な御意見，御助言を戴き，心から感謝いたします．また同ゼミにおいて曽根研究室をはじめとする各研究室の皆様に活発な御討論をしていただき，誠にありがとうございました．

日々の研究におきましては，計算機環境を整えていただいた北陸先端科学技術大学院大学阿部亨助教授，東北大学情報処理教育センター大町真一郎助手，阿曽研究室沼田一成氏，後藤英昭氏に感謝いたします．また，同研究室成富敬助手，黒岩丈介氏，森大毅氏をはじめとする阿曽研究室の皆様に数々の御意見，御討論戴いたことに深くお礼申し上げます．

最後に本研究を行なう上で貴重な音声資料を提供していただいたATR人間情報通信研究所に感謝いたします．

参考文献

[1] 北, 川端, 斎藤：“HMM 音韻認識と拡張 LR 構文解析法を用いた連続音声認識”, 情報処理学会論文誌, **Vol.31**, No.3, pp. 817–823 (1990).

[2] K. F. Lee and W. Hon: “Large-vocabulary speaker independent continuous speech recognition using hmm”, Proc.ICASSP'88, pp. 123–126 (1988).

[3] 松尾, 石亀：“トポロジーの学習と音素モデル間の相互依存を考慮した HMM による音素認識”, 信学論, **J76-D-II**, No.9, pp. 1835–1842 (1993).

[4] J. Takami and S. Sagayama: “A successive state splitting algorithm for effecient allophone modeling”, Proc.ICASSP'92, pp. 2155–2164 (1992).

[5] 鷹見, 嵯峨山：“逐次状態分割法による隠れマルコフ網の自動生成”, 信学論, **J76-D-II**, No.10, pp. 2155–2164 (1993).

[6] X. D. Huang, K. F. Lee, W. Hon and M. Y. Hwang: “Improved acoustic modeling with the sphinx speech recognition system”, Proc.ICASSP'91, pp. 345–348 (1991).

[7] M. Y. Hwang and X. D. Huang: “Subphonetic modeling with markov states – senon”, Proc.ICASSP'92, pp. 33–36 (1992).

[8] 川端：“音声理解システム JUNO における構文制御”, 平 6 秋音講論集, 1-Q-5 (1994).

[9] 村上：“メモリ量および計算量を削減した Baum-Welch アルゴリズムの提案と言語モデルへの適用”, 平 6 秋音講論集, 1-Q-6 (1994).

[10] 鈴木, 牧野, 阿曽, 下平：“逐次状態分割法の高速化”, 平 6 東北連大, 1G-3 p.227 (1994).

[11] R. Schwartz, Y. Chow, O. Kimball, S. Roucos, M. Krasner and J. Makhoul: “Context-dependent modeling for acoustic-phonetic recognition of continuous speech”, Proc.ICASSP'85, pp. 1205–1208 (1985).

[12] K. F. Lee, S. Hayamizu, H. W. Hon, C. Huang, J. Swartz and R. Wiede: “Allophone clustering for continuous speech recognition”, Proc.ICASSP'90, pp. 749–752 (1990).

[13] 鈴木, 牧野, 阿曽, 下平：“音響類似性に基づく HMnet を用いた音素認識”, 信学技報, SP94-15 (1994).

[14] 鈴木, 牧野, 阿曽, 下平：“音響類似性に基づく HMnet を用いた子音認識”, 平 6 秋音講論集, 1-8-5 (1994).

[15] M. K. McCandless and J. R. Glass: “Empirical acquisition of language models for speech recognition”, Proc.ICSLP'94, pp. 835–838 (1994).

[16] 鷹見：“状態分割融合法による隠れマルコフ網の表現効率向上”, 平 6 秋音講論集, 1-8-4 (1994).

[17] 中川：“確率モデルによる音声認識”, 電子情報通信学会編 (1988).

研究業績

- 発表論文

1. “A New HMnet Construction Algorithm Requiring No Contexual Factors”
   Motoyuki Suzuki, Shozo Makino, Akinori Ito,
   Hirotomo Aso, Hiroshi Shimodaira.
   IEICE Trans. syst.&info. (掲載予定)

- 学会発表等

1. “環境を明に考慮しないSSSによるHMnetの不特定話者への適用に関する考察”
   鈴木 基之，牧野 正三，阿曽 弘具，下平 博.
   日本音響学会 平成6年度春季研究発表会，2–P–18，pp.193–194 (1994-3)
2. “音響類似性に基づくHMnetを用いた音素認識”
   鈴木 基之，牧野 正三，阿曽 弘具，下平 博.
   電子情報通信学会音声研究会，SP95-15，pp.9–14 (1994-6)
3. “逐次状態分割法の高速化”
   鈴木 基之，牧野 正三，阿曽 弘具，下平 博.
   平成6年度 電気関係学会東北支部連合大会，1G-3，p.227 (1994-8)
4. “音響類似性に基づくHMnetを用いた子音認識”
   鈴木 基之，牧野 正三，阿曽 弘具，下平 博.
   日本音響学会 平成6年度秋季研究発表会，1–8–5，pp.9–10 (1994-10)

付録 A

Baum-Welch の再推定アルゴリズム

Baum-Welch アルゴリズム[1]は，HMM のパラメータをサンプル系列から推定するアルゴリズムとしてよく用いられる．ここでは，その原理と具体的な推定式を説明する [17].

A.1 EM アルゴリズム

HMM のパラメータを最尤推定することは，出力ベクトル系列に対し状態遷移系列が直接観測できず，各状態遷移が何回生じたか数えられないので容易ではない．これは，観測データが不完全な場合の最尤推定と本質的に同じ問題を含んでいる．不完全な観測データ$\boldsymbol{y}$に対して尤度関数$L(\boldsymbol{\theta};\boldsymbol{y})$ を最大にするパラメータ$\boldsymbol{\theta}$を決定するために$\boldsymbol{y}$に付随して得られる付加データ$\boldsymbol{x}$を用いる場合を考える．$\boldsymbol{y}$と$\boldsymbol{x}$を用いて推定されたパラメータ$\tilde{\boldsymbol{\theta}}$と初期値$\boldsymbol{\theta}$に対して Baum は次の重要な定理を証明した．

$$L(\tilde{\boldsymbol{\theta}};\boldsymbol{y}) \geq L(\boldsymbol{\theta};\boldsymbol{y}) \tag{A.1}$$

ここで，等号が成立する必要十分条件は$\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}$である．故に，この$\tilde{\boldsymbol{\theta}}$をあらたに$\boldsymbol{\theta}$として$\tilde{\boldsymbol{\theta}}$の推定を繰り返せは，最適値または局所的最適値に収束することになる．この手法は EM (Expected Maximization) アルゴリズムとしてよく知られているものである．

A.2 Baum-Welch アルゴリズム

前節の EM アルゴリズムの考え方を HMM のパラメータ推定に適用する場合を考える．HMM では，観測されるデータ$\boldsymbol{y}$はモデルから出力されるベクトル系列，$\boldsymbol{y}$ に付随して得られる非観測データ$\boldsymbol{x}$は状態遷移系列である．以下に具体的アルゴリズムを述べる．

まず，出力ベクトル系列$\boldsymbol{Y} = \boldsymbol{y}_1, \boldsymbol{y}_2, \cdots \boldsymbol{y}_T$に対し，$\alpha(i,t), \beta(i,t)$ を定義する．$\alpha(i,t)$ は，HMM が$\boldsymbol{y}_1, \boldsymbol{y}_2 \cdots \boldsymbol{y}_t$を出力して状態 i に遷移する確率を表しており，以下の漸化式で計算される．

$$\alpha(i,0) = \pi_i \tag{A.2}$$

$$\alpha(i,t) = \sum_j \alpha(j,t-1) a_{ji} b_{ji}(\boldsymbol{y}_t) \tag{A.3}$$

[1]または，Forward-Backword アルゴリズムと呼ばれる

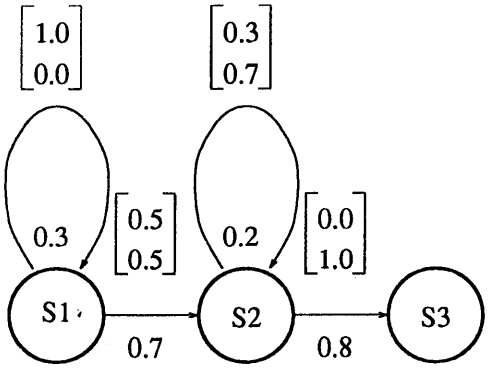


図 A.1: HMM の例．大括弧内の数字はベクトル $\boldsymbol{a}, \boldsymbol{b}$ を出力する確率

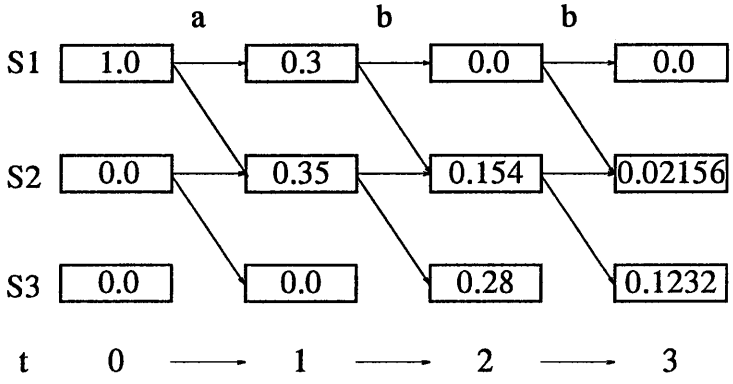


図 A.2: ベクトル系列 $\boldsymbol{aab}$ に対するトレリス

ここで，π_iは初期状態が i である確率，a_{ji}は状態 jから状態 i へ遷移する確率，$b_{ji}(\boldsymbol{y}_t)$ は状態 j から状態 i へ遷移する時にベクトル$\boldsymbol{y}_t$を出力する確率である．図 A.1がベクトル系列$\boldsymbol{abb}$を出力した時の$\alpha(i,t)$ の計算を図 A.2に示す．図 A.2はトレリスと呼ばれ，右下の数字がこの HMM がベクトル系列$\boldsymbol{abb}$ を出力する確率 $P(\boldsymbol{Y})$ を表している．また，式(A.3)で$\sum_j$を$\max_j$に変えたものをviterbi アルゴリズムと呼ぶ．このアルゴリズムを用いると状態遷移が一意に決まるので，バックトレースすることで状態遷移系列を決定することができる．

また$\beta(i,t)$ は$\alpha(i,t)$ と双対の関係にあり，HMM が状態 i を出発点として最終状態へと遷移していく間に$\boldsymbol{y}_{t+1}, \boldsymbol{y}_{t+2}, \cdots \boldsymbol{y}_T$を出力する確率を表している．

$$\beta(i,T) = \begin{cases} 1 & \text{if } \ i \in F \\ 0 & \text{otherwise} \end{cases} \tag{A.4}$$

$$\beta(i,t) = \sum_j \beta(j,t+1) a_{ji} b_{ji}(\boldsymbol{y}_{t+1}) \tag{A.5}$$

ここで，Fは最終状態の集合である．

定義から明らかなように，

$$\sum_{i\in F}\alpha(i,T)=\sum_{i}\beta(i,0)\pi_i=P(\boldsymbol{Y}) \tag{A.6}$$

となる．

これらを使って，パラメータを推定する．ここで，実際のパラメータ推定には多数のサンプル $\boldsymbol{Y}^1,\boldsymbol{Y}^2,\cdots\boldsymbol{Y}^N$ を用い，$P(\boldsymbol{Y}^1,\boldsymbol{Y}^2,\cdots\boldsymbol{Y}^N)=P(\boldsymbol{Y}^1)P(\boldsymbol{Y}^2)\cdots P(\boldsymbol{Y}^N)$ を最大にするように推定しなければならない．そこで，サンプル $\boldsymbol{Y}^n$ による $\alpha(i,t),\beta(i,t)$ を $\alpha^n(i,t),\beta^n(i,t)$ として，次のように推定する．ただし，$P_n=P(\boldsymbol{Y}^n)$ は，HMM がベクトル系列 $\boldsymbol{Y}^n$ を出力する確率である．

まず，初期状態確率の推定値 $\hat{\pi}_i$ は

$$\hat{\pi}_i=\frac{\sum_{n}^{N}\frac{1}{P_n}\sum_{j}\gamma^n(i,j,1)}{\sum_{n}^{N}\frac{1}{P_n}\sum_{i,j}\gamma^n(i,j,1)} \tag{A.7}$$

ここで，

$$\gamma^n(i,j,t)=\frac{\alpha^n(i,t-1)a_{ij}b_{ij}(\boldsymbol{y}_t^n)\beta^n(j,t)}{\sum_{i\in F}\alpha^n(i,T^n)} \tag{A.8}$$

である．また，状態遷移確率の推定値 $\hat{a}_{ij}$ は，

$$\hat{a}_{ij}=\frac{\sum_{n}^{N}\frac{1}{P_n}\sum_{t}\alpha^n(i,t-1)a_{ij}b_{ij}(\boldsymbol{y}_t^n)\beta^n(j,t)}{\sum_{n}^{N}\frac{1}{P_n}\sum_{t}\alpha^n(i,t)\beta^n(i,t)} \tag{A.9}$$

となる．出力確率の推定値 $\hat{b}_{ij}(\boldsymbol{y})$ は，出力確率分布の形状が離散分布の場合と，連続分布の場合があり，それぞれ以下のように推定する．

(1) 離散分布の場合

離散分布の場合は，状態 i から状態 j へ遷移する時に出力シンボル k を出力する確率の推定値 $\hat{b}_{ij}(k)$ を，以下のように推定する．

$$\hat{b}_{ij}(k)=\frac{\sum_{n}^{N}\frac{1}{P_n}\sum_{t:y_t^n=k}\alpha^n(i,t-1)a_{ij}b_{ij}(y_t^n)\beta^n(j,t)}{\sum_{n}^{N}\frac{1}{P_n}\sum_{t}\alpha^n(i,t-1)a_{ij}b_{ij}(y_t^n)\beta^n(j,t)} \tag{A.10}$$

(2) 連続分布の場合

出力確率分布が連続分布の場合，分布の形状の選び方で推定式は変ってくる．ここでは，現在最もよく用いられている分布形状である，各次元無相関のガウス分布であるとして推定式を導く．

各次元無相関のガウス分布であるので，ガウス分布の平均ベクトルを$\boldsymbol{\mu}_{ij}$，分散ベクトルを$\boldsymbol{\sigma}^2_{ij}$とすると，確率密度関数は，

$$b_{ij}(\boldsymbol{y}) = \prod_k^K \left(\frac{1}{2\pi\sigma^2_{ijk}}\right)^{\frac{1}{2}} \exp\left\{-\sum_k^K \frac{1}{2\pi\sigma^2_{ijk}}(y_k - \mu_{ijk})^2\right\} \tag{A.11}$$

と表せる．ここで，Kは次元数である．

この時，推定値$\hat{\boldsymbol{\mu}}_{ij}(\boldsymbol{y}), \hat{\boldsymbol{\sigma}}^2_{ij}$は，

$$\hat{\boldsymbol{\mu}}_{ij} = \frac{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t-1)a_{ij}b_{ij}(\boldsymbol{y}^n_t)\beta^n(j,t)\boldsymbol{y}^n_t}{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t-1)a_{ij}b_{ij}(\boldsymbol{y}^n_t)\beta^n(j,t)} \tag{A.12}$$

$$\hat{\sigma}^2_{ijk} = \frac{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t-1)a_{ij}b_{ij}(\boldsymbol{y}^n_t)\beta^n(j,t)(y^n_{tk} - \mu_{ijk})^2}{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t-1)a_{ij}b_{ij}(\boldsymbol{y}^n_t)\beta^n(j,t)} \tag{A.13}$$

となる．

特に，任意のjに対し$\boldsymbol{\mu}_{ji} = \boldsymbol{\mu}_i, \sigma^2_{ji} = \sigma^2_i$ である時，つまり，出力確率が状態遷移先の状態のみに依存する時は，

$$\hat{\boldsymbol{\mu}}_i = \frac{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t)\beta^n(i,t)\boldsymbol{y}^n_t}{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t)\beta^n(i,t)} \tag{A.14}$$

$$\hat{\sigma}^2_{ik} = \frac{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t)\beta^n(i,t)(y^n_{tk} - \mu_{ik})^2}{\sum_n^N \frac{1}{P_n}\sum_t \alpha^n(i,t)\beta^n(i,t)} \tag{A.15}$$

となる．

A.3 HMnet のパラメータ推定

HMnet は数多くの HMM の状態を共有させたものであり，学習サンプル１つ１つにとってみれは，HMM と同じである．そこで，パラメータ推定に Baum-Welch アルゴリズムを使うことができる．具体的には，各サンプル毎に$\alpha^n(i,t), \beta^n(i,t)$ を計算し，以下の式でパラメータの推定値を得る．なお，ここでは出力確率分布として無相関ガウス分布を仮定し，また状態遷移先の状態にのみ依存するものとする．

$$\hat{a}_i = \frac{\sum_{n\in S_i} \frac{1}{P_n}\sum_t \alpha^n(i,t-1)a_i b_i(\boldsymbol{y}^n_t)\beta^n(i,t)}{\sum_{n\in S_i} \frac{1}{P_n}\sum_t \alpha^n(i,t)\beta^n(i,t)} \tag{A.16}$$

$$\hat{\mu}_i = \frac{\sum_{n \in S_i} \frac{1}{P_n} \sum_t \alpha^n(i,t)\beta^n(i,t)\boldsymbol{y}_t^n}{\sum_{n \in S_i} \frac{1}{P_n} \sum_t \alpha^n(i,t)\beta^n(i,t)} \tag{A.17}$$

$$\hat{\sigma}_{ik}^2 = \frac{\sum_{n \in S_i} \frac{1}{P_n} \sum_t \alpha^n(i,t)\beta^n(i,t)(y_{tk}^n - \mu_{ik})^2}{\sum_{n \in S_i} \frac{1}{P_n} \sum_t \alpha^n(i,t)\beta^n(i,t)} \tag{A.18}$$

ここで，S_iは，状態 i を通る学習サンプルの集合であり，また a_iは状態 i で自己ループする確率である．

このようにして，HMM と同様に HMnet のパラメータを再推定することができる．

本審査 OHP 資料:

音響類似性に基づく隠れマルコフ網を用いた連続音声認識に関する研究

東北大学大学院工学研究科
電気及通信工学専攻
鈴木 基之

第１章 序論

音声認識技術の向上
　→自然なマン・マシンインターフェースを実現

現状... 単語発声程度ならば，実用化レベル

- ビデオの予約
- 切符の自動販売機

より自然な発声の認識
⇒ 連続音声を対象とした高精度な
　認識システム (音素認識部，言語処理部) が必要

○従来からの音声認識手法

- Hidden Markov Model(HMM)
- DP によるテンプレートマッチング
- Neural Network

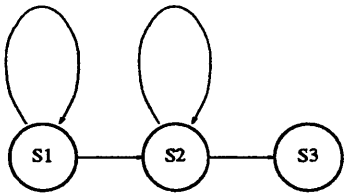
→HMM を基本とするものが主流

◎ HMM の利点

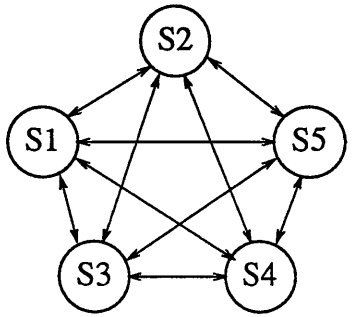
- 発声のゆらぎなどを統計的に扱える
- 多量のサンプルから自動学習できる
- 音声とモデルの対応が比較的明確

☆ HMM の形状

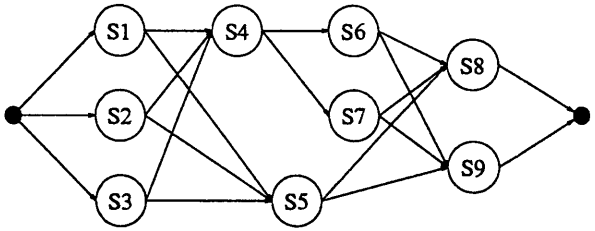
- left-to-right



- ergodic



- HMnet



→ HMnet を音素，言語のモデルに適用

◎本論文の目的

HMnet をモデルとし，連続音声を対象とした高精度な音声認識手法を開発

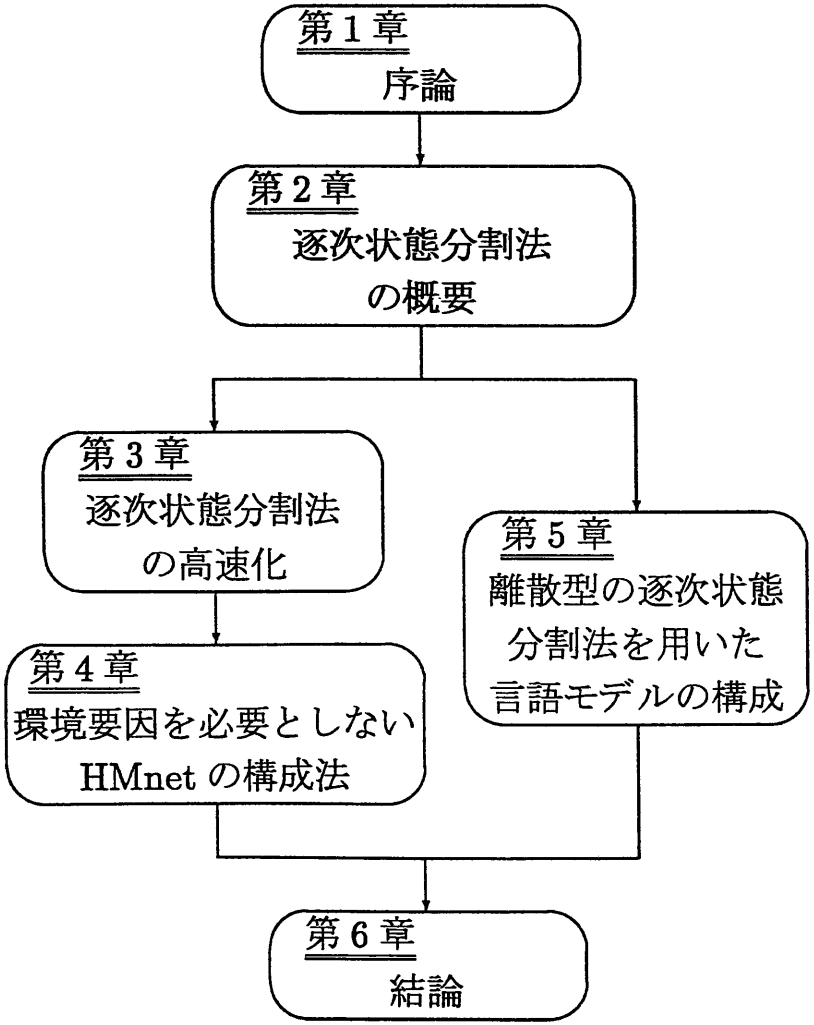
- 音素認識部

→環境要因を必要としないコンテキスト依存モデル構成法の提案

- 言語処理部

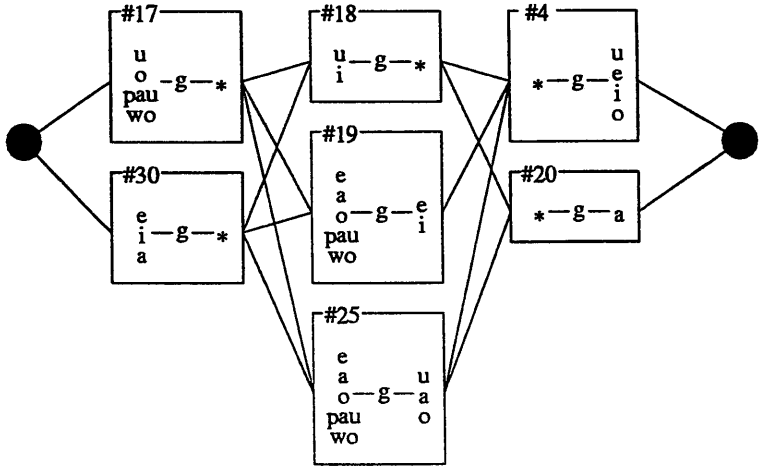
→離散型 HMnet 構成法の提案

☆本論文の構成



第2章 逐次状態分割法の概要

概要 小規模なモデルを初期モデルとし，出力分布の分散の拡がった状態を逐次分割していく，top down 的な手法.



☆特徴

- 時間方向も含めて，構造を自動的に決定
- 計算量が，通常の HMM に比べて膨大

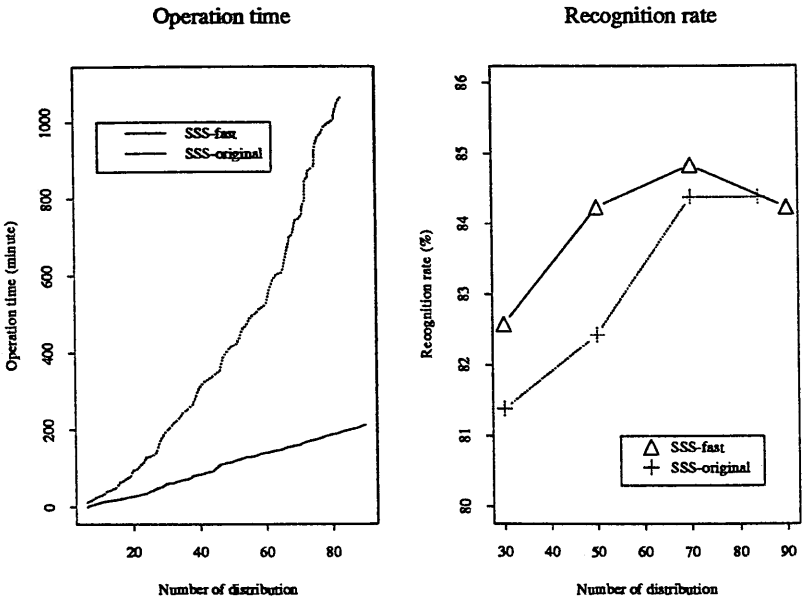
第3章 逐次状態分割法の高速化

逐次状態分割法の計算が遅い原因

- 状態が増えるたびに再推定をする
- 出力確率分布が混合数2の分布

→1ループあたりの計算量削減

局所的に混合数2の分布を用いる



認識率同程度で，計算時間が約1/5(状態数80)

第4章 環境要因を必要としない HMnetの構成法

☆コンテキスト依存モデルの本質的問題

⇓

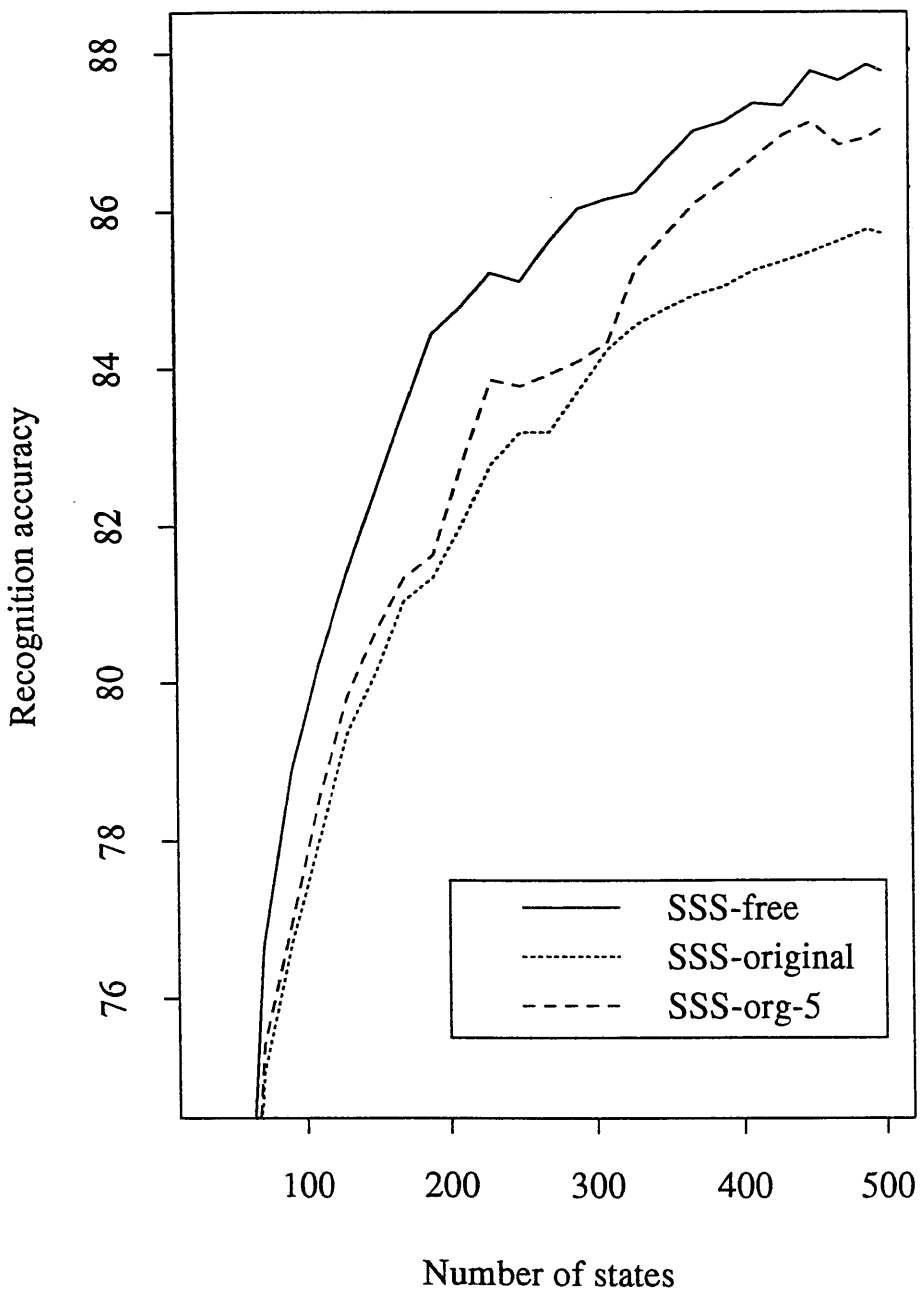
考慮する音素環境の要因を
先験的知識に基づいて決定

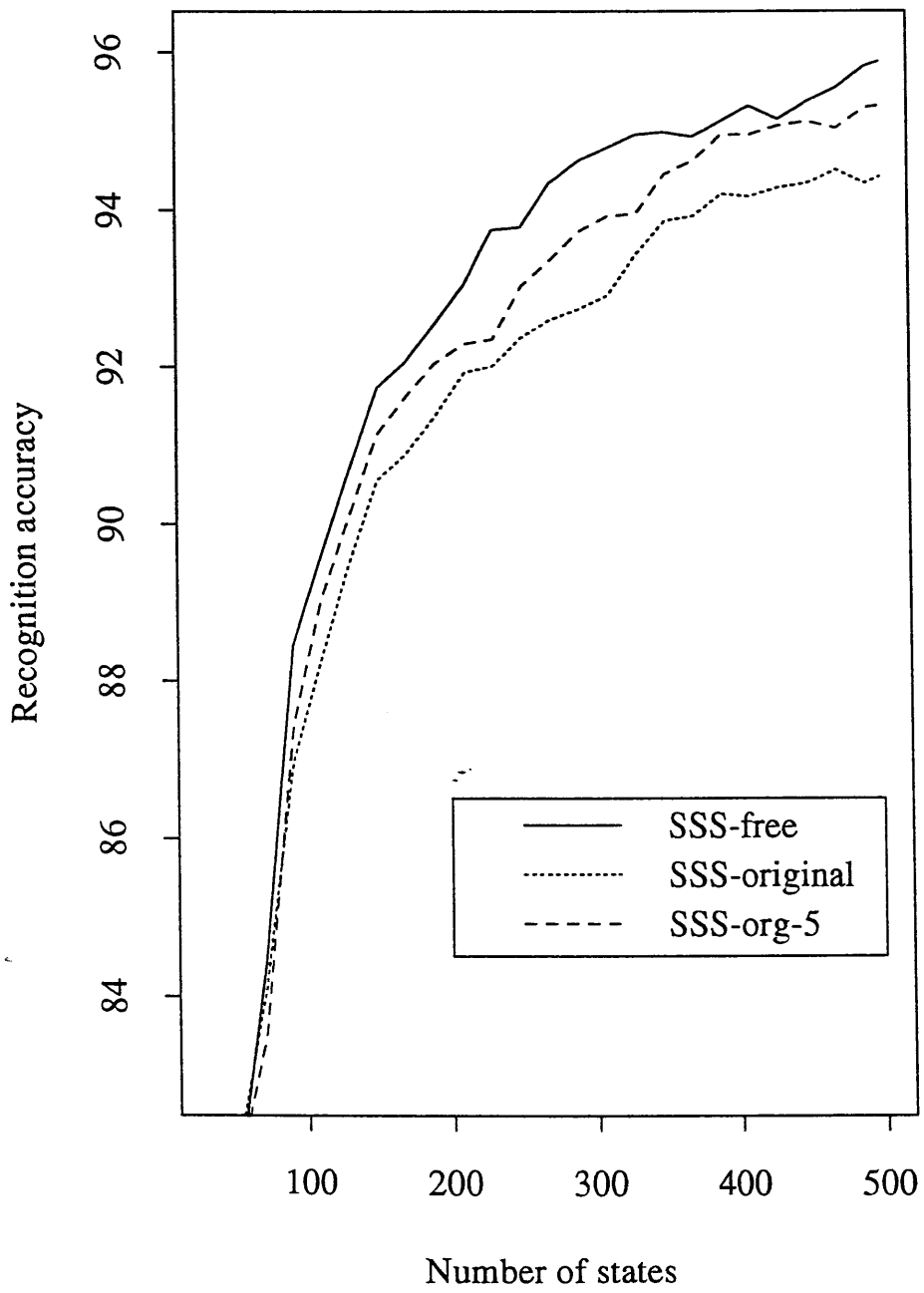
音響類似性に基づく分割

学習サンプル1つ1つについて，尤度を基準に分割

⇓

一番影響の大きい要因について分割
自動的によいモデルを構成


speaker MHO
Recognition accuracy
76
78
80
82
84
86
88
100
200
300
400
500
Number of states
SSS-free
SSS-original
SSS-org-5


speaker MTK
Recognition accuracy
84
86
88
90
92
94
96
100
200
300
400
500
Number of states
SSS-free
SSS-original
SSS-org-5

第5章 離散型逐次状態分割法を用いた言語モデルの構成

☆言語モデル

- 文法的な知識を用いるもの
  - 文法を人が与える必要有
  - タスクをよく表すモデルになる
- 統計的なモデル
  - 十分な学習サンプルが必要
  - 自動的に文法を獲得

→よりよい統計的モデルを開発

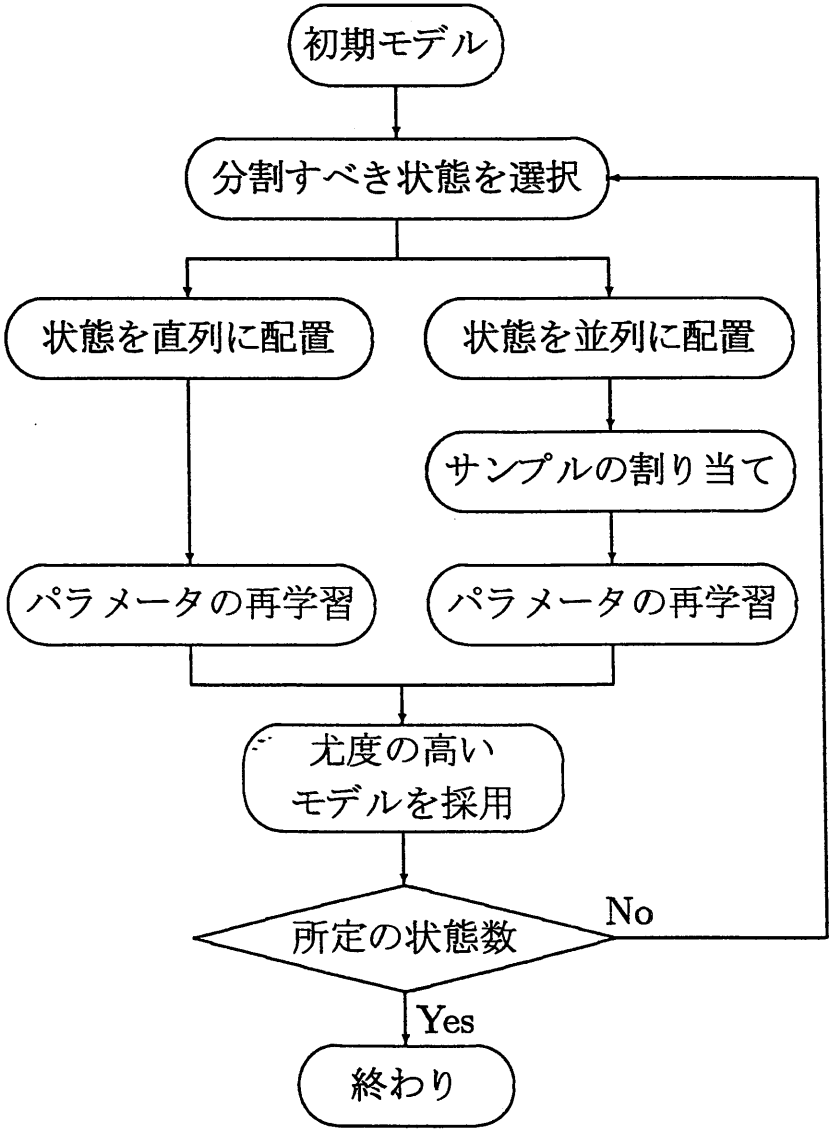
○従来の統計的言語モデル

- n-gram
- ergodic HMM

} → 時間的な記述能力に欠ける

⇒HMnet を言語モデルに適用

☆離散型逐次状態分割法



◎実験した手法

1. 分割する状態の選択

(a) 出力分布のエントロピー

$$V_i = n_i \times \sum_k \{-P_i(k) \log P_i(k)\}$$

(b) 単語間距離による歪み

$$V_i = n_i \times \min_j \sum_k \{Wd(j,k) \times P_i(k)\}$$

ここで,

$$Wd(i,j) = K(Pp_i, Pp_j) + K(Pn_i, Pn_j)$$

$$K(P,Q) = \sum_j \left\{ P(j) \log \frac{P(j)}{Q(j)} + Q(j) \log \frac{Q(j)}{P(j)} \right\}$$

2. クラスタリング手法

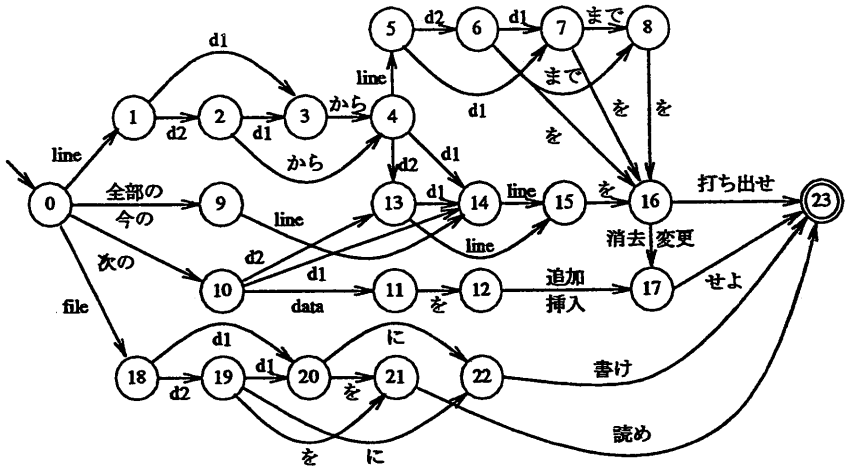
→最大距離クラスタリング法

3. サンプル間の距離尺度

(a) Kullback Divergence

(b) DP 距離

☆認識対象言語



※単語数 36. 次の状態への遷移確率は等しい.

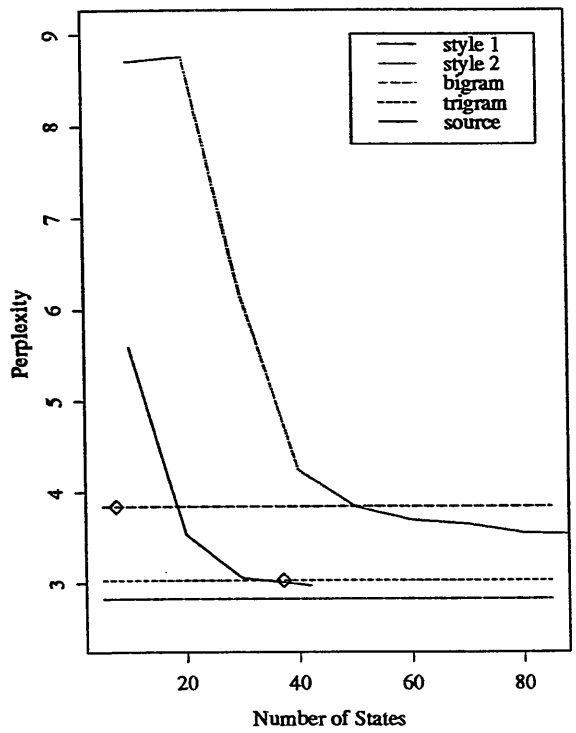
学習サンプル 30000 文

テストサンプル 10000 文

評価基準 perplexity

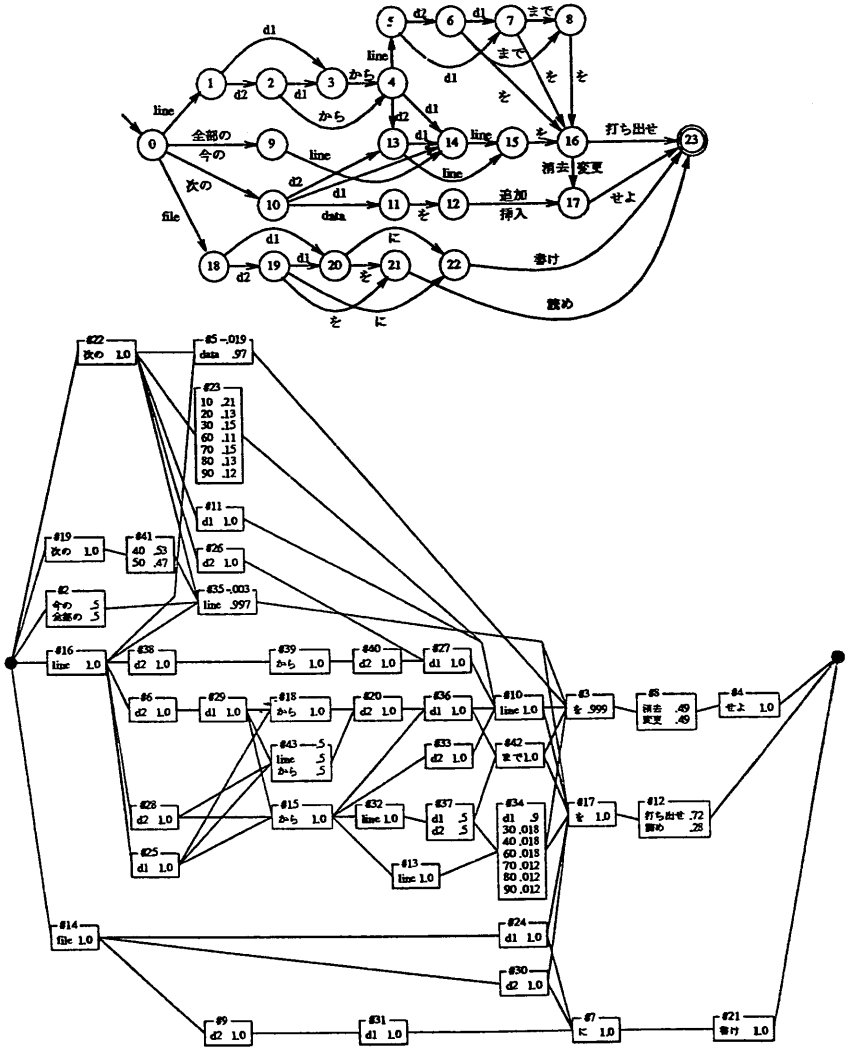
※ perplexity=情報理論的な意味での平均分岐数.

Perplexity



※ bigram と trigram は，それぞれ◇の位置における状態数の HMnet と同数のパラメータを持つ.

☆得られた HMnet の構造



第5章のまとめ

HMnet を言語モデルに適用

- 単語間距離を考慮した手法で trigram を越える性能
- 真の文法とほぼ同様な文法構造を獲得

◎今後の予定

- 距離尺度などの再検討
- 自然言語への適用

第6章 結論

HMnet をモデルとして，連続音声の高精度な認識手法を提案

- 逐次状態分割法を高速化
  →認識率同程度で計算時間は 1/5 程度 (80 状態)
- 音素の変形要因を必要としないモデル構成法を提案
  →音響的特徴をよく表現した HMnet
  音素の誤認識率が 21%程度改善
- 離散型の逐次状態分割法を提案し，言語モデルを構成
  →trigram モデルを越える性能

☆今後の予定

音素認識部と言語処理部を融合した，一体型の連続音声認識システムの開発