TR-IT-0244

ニューラルネットワークと言語統計量に基づく発音辞書の自動生成

Automatic Generation of Multiple Pronunciations Based on Neural Networks and Language Statistics

吉村 貴克
Takayoshi Yoshimura

深田 俊明
Toshiaki Fukada

1997 年 12 月 5 日

自然発話音声では，読み上げ発声では起こらないような，大きな発声変形を生じることがある．このような発声を音声認識しようとした場合，標準的な読みが付与された発音辞書を用いても，正しい認識結果は得られない．つまり，標準的な発音系列と実際に発声される発音系列のミスマッチを緩和する機構が必要である．本報告では，既に提案した発音ネットワークに基づく発音辞書の自動生成 の方法を発展させ，(1) 複数発音に尤度を付与，(2) 言語統計量を利用した発音辞書の生成を行うことにより，先に提案した方法よりも，更に優れた発音辞書が生成できることを示す．

目次

1 はじめに

話者，単語，コンテクストなどの違いによる発声変形を考慮した発音辞書の構築は，不特定話者音声認識システムにおける認識制度の向上を実現する上で重要な課題である．発音辞書の構築には，ルールによるものや，人手によるものが考えられるが，ルールによる発音辞書の作成は，エキスパートを必要とし，実際の発声変形に対する正確なモデリングができている保証がないという問題がある．人手による発音辞書の作成は，作業時間が膨大になるという問題がある．このため，データベースを利用して，人手を介することなく発音辞書を自動的に生成しようとする試みも近年数多く検討されてきている [1][2].

本報告では，既に提案した発音ネットワークに基づく発音辞書の自動生成 [3] の方法を更に発展させ，(1) 複数発音に尤度を付与，(2) 言語統計量を利用した発音辞書の生成を行い，音声認識実験により，先に提案した方法よりも，更に優れた発音辞書が生成できることを示す．

2 ニューラルネットワークと言語統計量を利用した発音辞書の自動生成

2.1 発音ネットワークの作成

標準的な発音系列から実際に発声される発音系列を予測するモデルとして，発音ネットワークを以下の手順で作成する．

1. 学習データに対して音素認識を実行し，認識結果の音素系列を修正発音列とする．

2. 書き起こし読み系列 (標準発音列) と修正発音列との間で文字列レベルの DP をとる．例えば，「あらゆる (標準発音列 /a r a y u r u/, 修正発音列 /a w a u r i u/)」が，
   a　*r*　a　*y*　u　*r*　　u　(標準発音列)
   a　*w*　a　　u　*r*　*i*　u　(修正発音列)
   と対応付けられた場合，標準発音列 /a/ は修正発音列 /a/ になり (/a/ に置換と考える)，/r/ は /w/ に置換し，/a/ は /a/ に，/y/ は脱落となり，/u/ は /u/ に，/r/ は /r i/(/i/ が挿入) に，/u/ は /u/ になるとする．

1 の音素認識の結果を，表 2.1に示す．挿入が 2 個以上の音素は，全音素に対し 1% に満たないので，今回は無視した．

表 1: 音素認識の結果

全音素数	119584
置換	113339
挿入 1 個	2942
挿入 2 個	814
挿入 3 個	135
挿入 4 個	51
挿入 5 個	13
挿入 6 個	6
挿入 16 個	1
脱落	6245

発音ネットワークは，図 1に示す構造をもつ multi-layer perceptron 型のニューラルネットワークを用いて構築した．入力は，前後 2 音素ずつのコンテキストを考慮した 5 音素からなる標準発音列 $L(m-2),\ldots,L(m+2)$ であり，出力は，中心音素 $L(m)$ に対応する修正発音列 $A(m)$ である．ここで，入力層 130 ユニット (先々行音素 26, 先行 26, 当該 26, 後続 26, 後々続 26) 出力層 53 ユニット (置換 26, 挿入 26, 脱落 1) とした．また，標準発音列と修正発音列が一致している音素，例えば，/a/ が /a/ に対応している場合，/a/ が /a/ に置換したとして利用した．学習時の入力層と出力層のデータは，該当するユニットに 1 を与え，それ以外には 0 を与えた．例えば，標準発音列 /r a y u r/ ，および修正発音列 " "(脱落) の場合，入力層 $L(m-2)$ の /r/，$L(m-1)$ の /a/，$L(m)$ の /y/，$L(m+1)$ の /u/，$L(m+2)$ の /r/ のユニットにそれぞれ 1 を与え，それ以外のユニットには 0 を与える．このとき，出力層の脱落のユニットに 1 を与え，それ以外のユニットには 0 を与える．

2.2 複数発音の尤度付与

単語 W に対する複数発音は，発音ネットワークの出力値に基づいて上位 N 位候補として得ることができる．文献 [3] では，これら N 個の複数発音をすべて同等 (等確率) に利用していたが，ここでは，1 位候補のスコア値で正規化することにより，複数発音に尤度を与えることを考える．

単語 W の発音 Prn に対する正期化尤度を $P(Prn|W)$ とすると，音響パラメータの観測系列 $\mathbf{x}$ に対する単語 W の尤度 $P(\mathbf{x}|W)$ は，

$$P(\mathbf{x}|W) = P(\mathbf{x}|Prn) \cdot P(W) \cdot P(Prn|W) \tag{1}$$

で与えられる．ここで，$P(\mathbf{x}|Prn)$ は音素系列 Prn から得られる $\mathbf{x}$ に対する音響尤度であり，音響モデルから求められる．$P(W)$ は単語 W が生起する発音尤度であり，発音ネットワークから求めることができる．音声認識シ

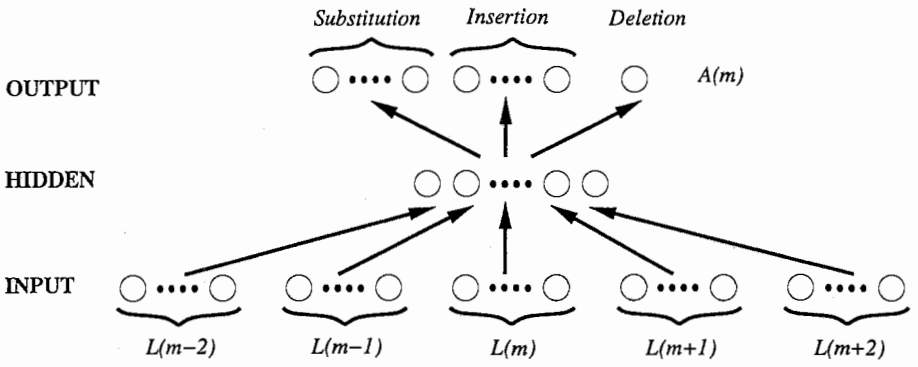

図 1: 発音ネットワークの構造

ステムでは，通常，言語尤度に重み α を適用することが多く，発音尤度に対しても同様の重み β を適用すると，式 (1) は，

$$P(\mathbf{x}|W) = P(\mathbf{x}|Prn) \cdot P(W)^{\alpha} \cdot P(Prn|W)^{\beta} \tag{2}$$

と書ける．ここで，発音ネットワークを適用しなかった 1 音素から 4 音素単語に対する $P(Prn|W)$ は 1.0 とする．

複数発音は，N により個数として与えることもできるが [3]，ここでは，正規化尤度の値がある閾値以上のものをすべて利用することにする．

2.3 言語統計量の利用

文献 [3] では，単語境界における音素 (M 音素からなる単語に対する $L(1), L(2), L(M-1), L(M)$) に対しては，先行あるいは後続の音素は辞書作成時は未知であるため発音ネットワークを適用していなかった．理想的には，認識時の仮説に応じてダイナミックに発声変形を行なえばよいが，実現は困難であると考えられる．発声変形を考慮しない辞書を用いて認識を行い，N ベスト出力に対して単語境界を考慮した発声変形を行うこともできるが [2]，ここでは単語バイグラムを利用することにより，辞書作成時に単語内の全音素に対して発声変形を考慮することを考える．

単語 W_C (音素数 M_{W_C}) の 1 番目音素 $L_{W_C}(1)$ に対する発声変形を考慮する場合，単語 W_C の先行単語 W_P (音素数 M_{W_P}) の後ろ 2 音素 $L_{W_P}(M_{W_P}-1)$, $L_{W_P}(M_{W_P})$ と，W_C の 1 番目から 3 番目までの音素 $L_{W_C}(1)$, $L_{W_C}(2)$, $L_{W_C}(3)$ のそれぞれに対応するユニットに 1 を入力する．このとき，i 番目の出力ユニットの出力スコアを $S_{W_C,i}(1)$，W_P から W_C が生起するバイグラムを $P(W_C|W_P)$ とすると，単語の全集合 $\mathbf{W}$ に対する W_C の 1 音素目のユニット i の出力スコアの期待値 $\bar{S}_{W_C,i}(1)$ は，

$$\bar{S}_{W_C,i}(1) = \sum_{W_P \in \mathbf{W}} P(W_C|W_P) S_{W_C,i}(1) \tag{3}$$

で与えられる．同様に，$2, M_{W_C}-1, M_{W_C}$ 番目の音素に対しても，単語バイグラムを利用して統計的に発声変形を考慮することができる．ここで，先行単語が 1 音素単語である場合は，先行単語の前の音素を /a,i,u,e,o,ng,-(無音)/ の 7 通りに対して発声変形を考慮し，出力スコアを等確率で分配し，後続単語が 1 音素単語である場合は，後続単語の次の音素をすべての音素セットの 26 通りを等確率で分配した．

2.4 発音尤度の修正

式 (2) における $P(Prn|W)$ は，正規化尤度としてそのまま利用することができるが，$P(Prn|W)$ の信頼度は，(1) 5 音素を確定して発音ネットワークから求めたもの，(2) 言語統計量をてきようしたもの，(3) 標準系列をそのまま適用 (1 音素から 4 音素の単語) したものの順に低くなっていくと考えられる．そこで，今回は，$P(Prn|W)$ に信頼度 k

$$k = \frac{\sum_{m=1}^{M} \prod_{i=-2}^{2} w(m-i)}{M} \tag{4}$$

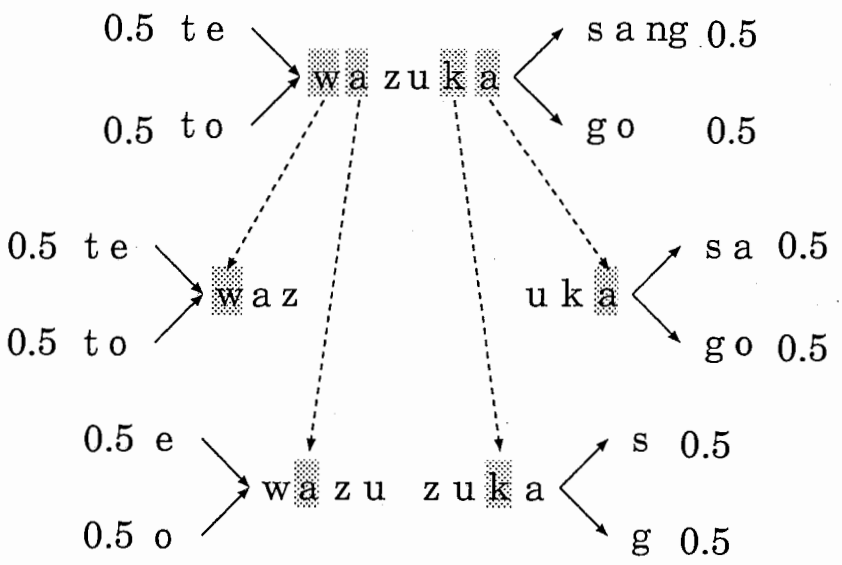

図 2: 言語統計量を利用した例 /w a z u k a/ (図中の数字はその単語が先行単語もしくは後続単語になる確率)

を乗じる．ここで，$w(m-i)$ は，表 2のような値をとる．

信頼度 k を乗じた修正発音尤度 $P'(Prn|W)$ を，

$$P'(Prn|W) = k \cdot P(Prn|W) \tag{5}$$

と定義し，この値を式 (2) の $P(Prn|W)$ に適用する．

表 2: $w(m-i)$ の値

$\|i\|$	0	1	2
音素既知	1.0	1.0	1.0
言語統計量適用	-	0.8	0.9
音素未知	-	0.7	0.8

3 実験

辞書を評価するために，Travel Arrangement をタスクとする自由発声音声データベース [4] に対して，単語グラフに基づく自然発話音声認識システム [5] を用いた認識実験を行った．

3.1 実験条件

発音ネットワークおよび音響モデルの学習には，230 名 (男性 100 名，女性 130 名，約 200 分) の音声を用い，認識実験のための評価データには，42 名 (男性 17 名，女性 25 名，約 40 分) を用いた．特徴パラメータは，標本化周波数 16kHz の音声データから求めた 12 次の MFCC と対数パワー，およびこれらの一次回帰係数の合計 26 次元とした．音響モデルは，ML-SSS アルゴリズム [6] によって性別非依存の 800 状態各 5 混合の HMnet を用いた (連続音声認識用音響モデル Version 2.0,TR-IT-0241)．この音響モデルを用いた音素タイプライタ認識の認識率は 74.3% である．ニューラルネットワークの繰り返し回数は 500 回とした．認識実験における言語モデルとして，クラス数 1000 の品詞・単語可変調 n-gram [7] を用いた．標準発音列を発声変形させ，複数の修正発音列 (N) を生成したとき，その中に標準発音列が含まれていない場合は標準発音列の発音尤度を 0.01 とし，修正発音列に加えた．2.2 で述べた複数発音を生成する際の正規化尤度の閾値は 0.4 とした．2.2 における音響尤度に対する言語尤度と発音尤度の重み α, β は同じ値とした．

認識用辞書としては，標準発音列に基づく辞書 (従来辞書) と人手により修正を行った辞書 (エキスパート辞書) の 2 種類に対して提案法を適用した．従来辞書は，個々のエントリに対して 1 種類の標準的な読みが付与され，全てのエントリに対してポーズが接続しうるように記述されている．エキスパート辞書は，人手によりポーズの接続の可否，濁音化，音素の脱落，置換などの発声変形等を考慮した複数発音辞書である．

言語統計量を用いて発声変形させる際，先行単語に pause で終る単語がきたり，発声変形させる単語が pause で後続単語に接続することはほとんどないと考え，辞書を言語統計量を用いて発声変形させるときは，従来辞書の場合，

1. 単語末の pause を抜いた従来辞書で言語統計量を用いて発声変形させた発音辞書
2. 単語末の pause を抜いた従来辞書で言語統計量を用いずに発声変形させた発音辞書
3. (2) の辞書の単語末に pause を付けた発音辞書

の 3 つの発音辞書をマージし，1 つの発音辞書を作成した．エキスパート辞書の場合は，pause で終る先行単語は無視して発声変形を行い，単語が pause で終っているときは，単語の最後の 2 音素は発声変形しないようにした．

3.2 従来辞書での提案法の効果

従来辞書を用いて認識実験を行った．複数発音に尤度を付与せず，言語統計量を利用しないとき (提案法 1) の認識率を表 3に示す．図中の言語重み 8 : 20 は，2.2 で述べた音響尤度に対する言語尤度と発音尤度の重み α, β の値が探索の 1 パス目は 8，2 パス目は 20 ということを意味している．

従来辞書をそのまま用いたもの (従来法) と比べ，処理時間はほぼ同じで，認識率は 1.3% 向上した．

表 3: 発音ネットワークにより作成した辞書を用いたときの認識率

	言語重み	辞書サイズ	認識率 (%)	改善率 (%)	CPU(%)
従来法	8:20	7484	65.5	-	200.0
提案法 1	8:20	28663	66.8	3.7	207.0

(3.2.1) 複数発音に尤度を付与したときの効果

複数発音に尤度を付与したとき (提案法 2) の認識率を，表 4に示す．ここで，言語重みを提案法 1 より小さくしているのは，発音尤度 $P(Prn|W)$ に信頼度 k をかけると，言語重みを大きくするのと等しくなり，認識率が低下するためである．これについての詳細は (3.2.3) で述べる．

表 4から，発音尤度を付与したとき，複数発音を等確率で扱ったとき (提案法 3) と比較すると，処理時間が若干減り，認識率がやや向上していることがわかる．

表 4: 複数発音に尤度を付与したときの認識率

	発音尤度	言語重み	辞書サイズ	認識率 (%)	改善率 (%)	CPU(%)
提案法 2	yes	7:20	28663	68.8	9.5	239.0
提案法 3	no	7:20	28663	68.7	7.4	243.8

(3.2.2) 言語統計量を利用したときの効果

言語統計量を利用し，発音列の両端 2 音素を発声変形させたとき (提案法 4) の認識率を，表 5に示す．統計量を利用しないときに比べ，辞書のサイズは約 5000 増え，認識率はやや向上した．

また，提案法 4 に発音尤度を付与したとき (提案法 5)，認識率は更に向上することが期待されたが，結果は 提案法 4 とほぼ同じだった．しかし，表 3の従来法に対し，約 10% の誤り改善率が達成できた．

表 5: 複数発音に尤度を付与したときの認識率

	発音尤度	言語統計量	言語重み	辞書サイズ	認識率 (%)	改善率 (%)	CPU(%)
提案法 3	no	no	7:20	28663	68.7	7.4	243.8
提案法 4	no	yes	7:20	33742	67.1	4.5	265.1
提案法 5	yes	yes	7:20	33742	68.9	9.8	265.2

(3.2.3) 信頼度の効果

提案法 3 で言語重みをかえて認識実験を行った．認識結果を表 6に示す．提案法 1 と提案法 3a を比べると，CPU Time が減少し，認識率も低下していることがわかる．図 3 は，言語重みと CPU Time の関係をグラフにしたものである．この図から，提案法 3 の言語重みを 7.5:20 から 7:20 の間の値に設定すれば提案法 1 の CPU Time と同じになることがわかるが，今回は信頼度を用いるときは，言語重みを 7:20 に統一した．

表 6: 複数発音に尤度を付与したときの認識率

	発音尤度	信頼度	言語重み	辞書サイズ	認識率 (%)	CPU(%)
提案法 1	no	no	8:20	28663	66.8	207.0
提案法 3a	no	yes	8:20	28663	64.2	155.5
提案法 3b	no	yes	7.5:20	28663	66.1	188.1
提案法 3c	no	yes	7:20	28663	68.7	243.8

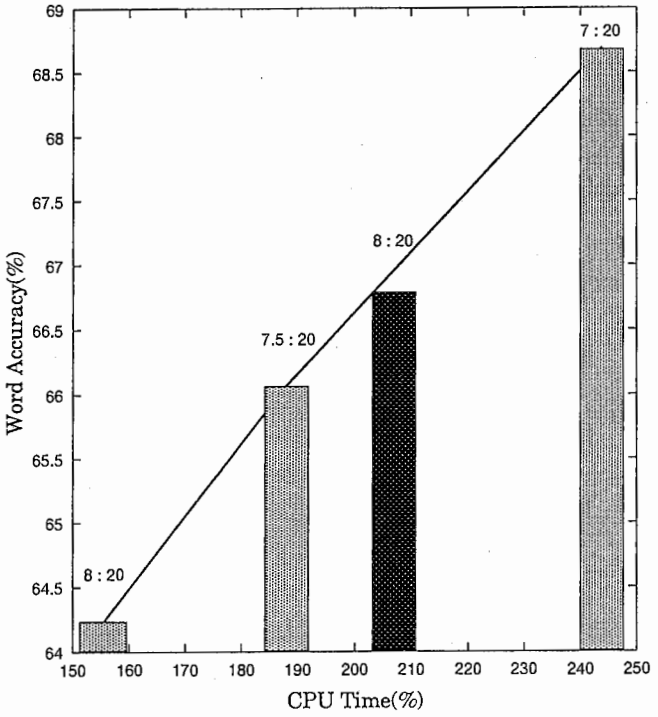

図 3: 言語重みと CPU Time の関係 (濃い棒グラフ: 提案法 1，薄い棒グラフ: 提案法 3)

3.3 エキスパート辞書での提案法の効果

エキスパート辞書を用いた認識実験の結果を，表7に示す．発音尤度のみを用いたとき，認識率が最も良くなった．発音尤度，言語統計量ともに用いたときは，発音尤度のみを用いたときと，ほぼ同じ認識率であった．

認識結果から提案法はエキスパート辞書にも効果があるといえる．

表 7: エキスパート辞書での提案法の効果

発音尤度	言語統計量	辞書サイズ	認識率 (%)	改善率 (%)	CPU(%)
-	-	7484	71.2	-	221.8
no	no	33198	72.1	3.7	231.2
no	yes	42103	72.6	5.4	263.3
yes	no	33198	74.0	10.2	266.5
yes	yes	42103	73.6	8.9	294.2

3.4 従来辞書とエキスパート辞書の比較

従来辞書とエキスパート辞書との認識率の違いを，図4に示す．図4の提案法は発音尤度と言語統計量を用いたときである．この図から，提案法は従来辞書，エキスパート辞書ともに，認識率が向上しており，また，従来辞書にくらべ，エキスパート辞書のほうが，従来法，提案法ともに認識率が上回っていることがわかる．

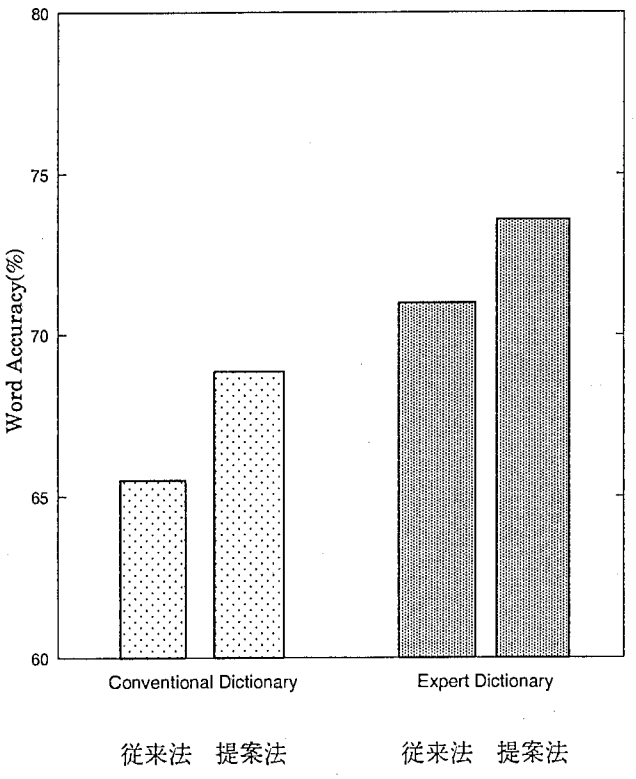

図 4: 従来辞書とエキスパート辞書の比較

4 結論

ニューラルネットワークに基づく発音辞書の自動生成法について述べた．ニューラルネットワークの出力値を発音尤度として利用すること，言語統計量として単語バイグラムを用いて単語境界の音素に対しても発声変形を考慮することにより，先に我々が提案した生成法の認識性能を上回ることを確かめた．また，提案法は，従来辞書の性能改善のみならず，エキスパートによって発声変形を考慮して作成された辞書の性能も更に改善できることも示した．

今後の可能性として，複数発音辞書を用いて学習データのアライメントを行うことにより，音響モデルを高精度化すること，発音ネットワークに品詞などの情報を付加し，より正確な発声変形を行うことなどが考えられる．

5 謝辞

本研究を進めるにあたり，ニューラルネットの学習プログラムのプログラムを提供して下さったMike Schuster研究員，言語モデルを提供して下さった雅瀧 浩和 研究員に深く感謝致します．また，御指導，御支援いただいた，匂坂 芳典 室長を始めとするATR音声翻訳通信研究所第一研究室の皆様に深く感謝致します．さらに実務訓練の機会を与えて下さった名古屋工業大学 知能情報システム学科の北村 正 教授，徳田 恵一 助教授及びATR音声翻訳通信研究所の山本 誠一 社長に心から感謝致します．

参考文献

[1] J. Humphries, P. Woodland : "Using accent-specific pronunciation modelling for improved large vocabulary continuous speech recognition," *Proc. EUROSPEECH-97*, pp. 2367–2370, 1997.

[2] M. Weintraub, E. Fosler, C. Galles, Y.-H. Kao, S. Khudanpur, M. Saraclar, S. Wegmann : "Automatic learning of word pronunciation from data," *JHU Workshop-96 Project Report*, 1996.

[3] 深田 俊明, 匂坂 芳典: "発音ネットワークに基づく発音辞書の自動生成" , 信学論 (D-II), **J80-D-II**, 10 pp. 2626–2635 (1997-10).

[4] A. Nakamura, S. Matsunaga, T. Shimizu, M. Tonomura and Y. Sagisaka: "Japanese speech databases for robust speech recognition," *Proc. ICSLP-96*, pp. 2199–2202, 1996.

[5] 清水 徹, 山本 博史, 政瀧 浩和, 松永 昭一, 匂坂 芳典: "大語い連続音声認識のための単語仮説数削減," 信学論 (D-II), **J79-D-II**, 12 pp. 2117–2124 (1996-12).

[6] M. Ostendorf and H. Singer: "HMM topology design using maximum likelihood successive state splitting," *Computer Speech and Language*, 11, pp. 17–41, 1997.

[7] H. Masataki and Y. Sagisaka : "Variable-order n-gram generation by word-class splitting and consecutive word grouping," *Proc. ICASSP-96*, pp. 188–191, 1996.

付録 A　NN の学習の繰り返し回数を変えたときの認識率

NN の学習の繰り返し回数を変え，発音辞書を作成し認識実験を行った．実験条件は，発音尤度の threshold は 0.5 で，それ以外は 3.1と同じである．辞書はエキスパート辞書を用いた．結果を図 5に示す．図 5認識率は繰り返し 500 回付近でピークに達し，500 回以降はほとんど変わらないことがわかる．また，繰り返し回数がいくつであっても，認識率は baseline を越えていることがわかる．

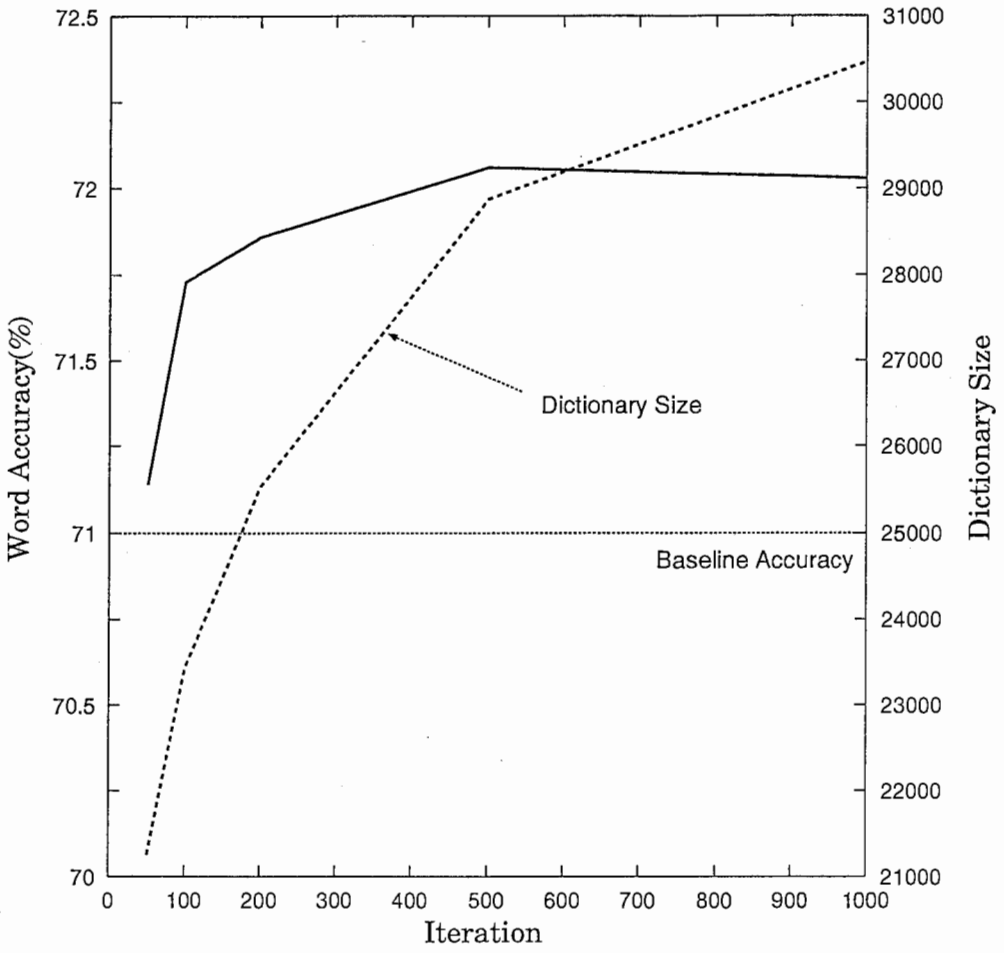

図 5: NN の学習の繰り返し回数を変えたときの認識率

付録 B　発音辞書の発音尤度の threshold を変えたときの認識率

発音尤度の threshold を変え，認識実験を行った．実験条件は threshold 以外は 3.1 と同じで，辞書はエキスパート辞書を用いた．図 6は，複数の修正発音系列に C(標準発音系列) を加えた場合，図 7は，複数の修正発音系列に C(標準発音系列) を加えない場合の認識率である．2 つの図から，どちらの場合も threshold 0.4 付近でピークになっていることがわかる．また，標準発音系列を加えた場合は，threshold 1.0 から 0.2 の間では baseline の認識率を越えており，一方，標準発音系列を加えない場合は threshold が大きいときは，baseline の認識率を下回ることがわかる．

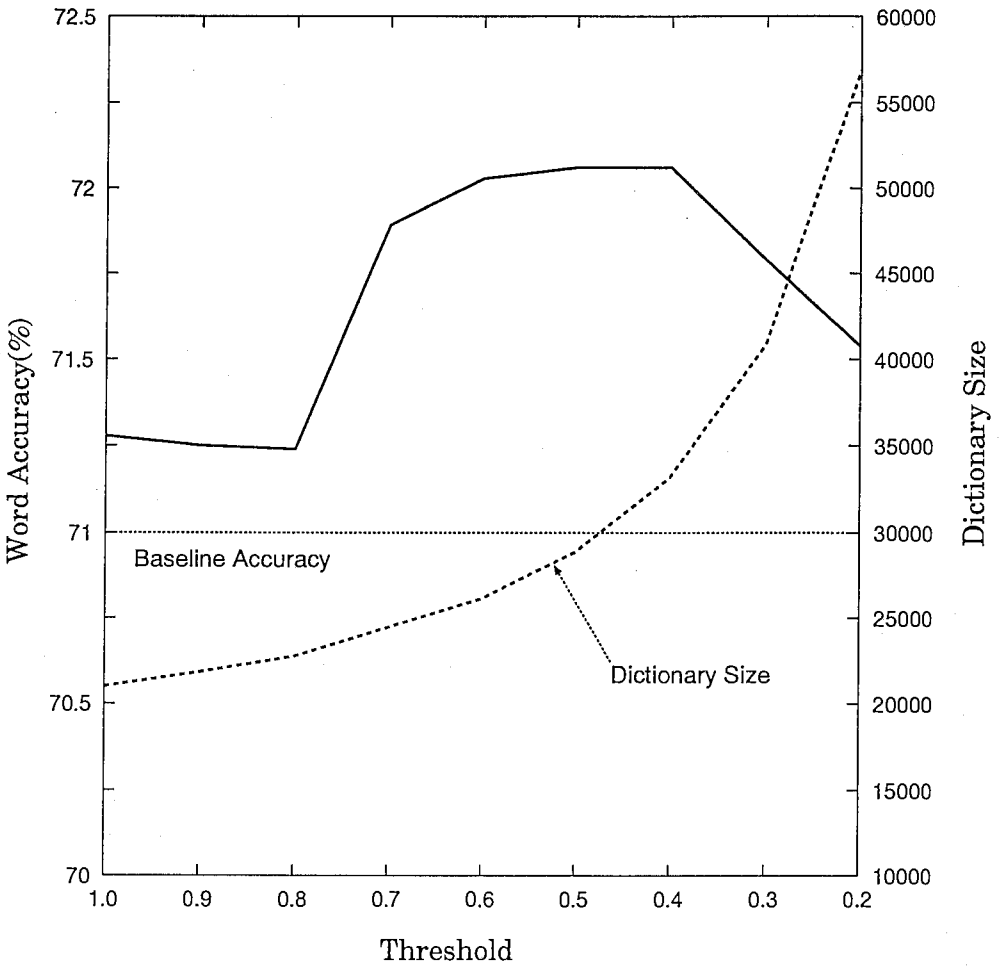

図 6: C(標準発音系列) を加えたとき

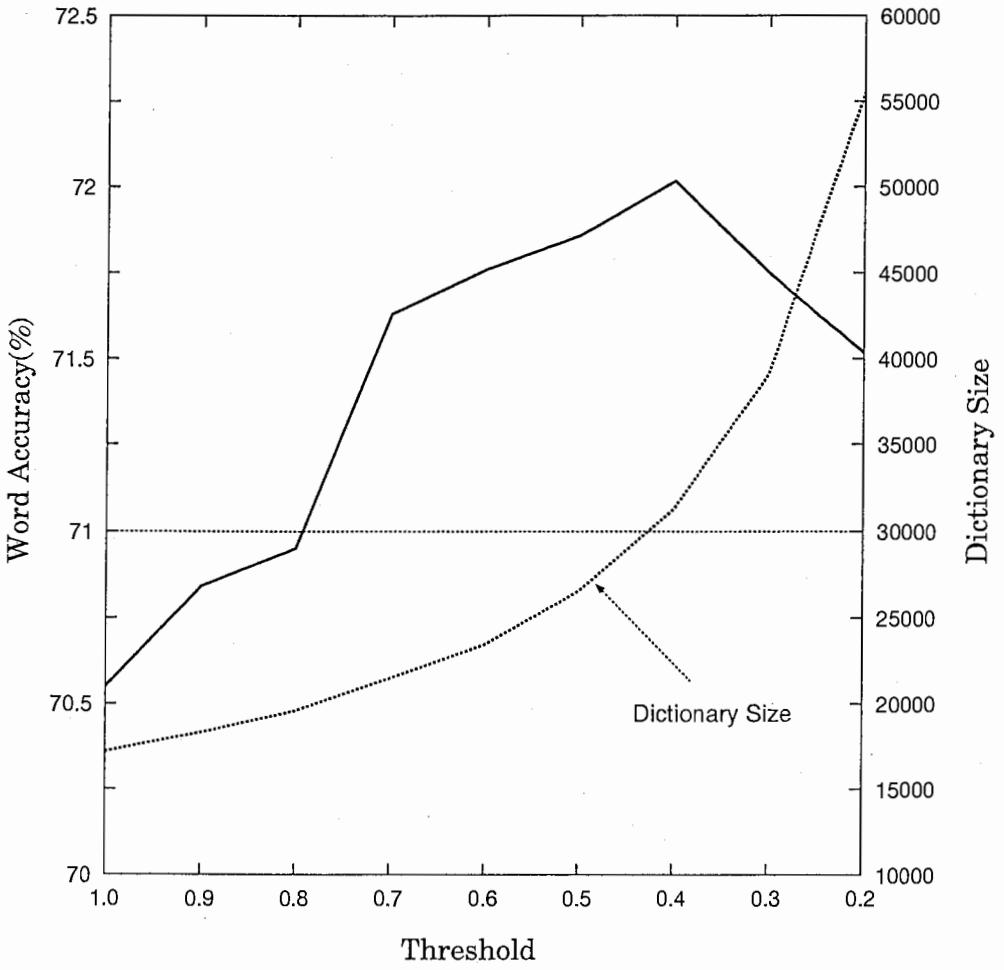

図 7: C(標準発音系列) を加えないとき

付録 C　音響尤度に対する発音尤度の重み β の値を変えたときの認識率

発音尤度のみ用いて，発音尤度の重み β の値を 1,2,4,8 と変化させ，認識実験を行った．実験条件は 3.1と同じで，辞書は従来辞書を用いた．結果を図 8に示す．発音尤度の重み $\beta = 1$ のとき，認識率が最も大きくなっている．

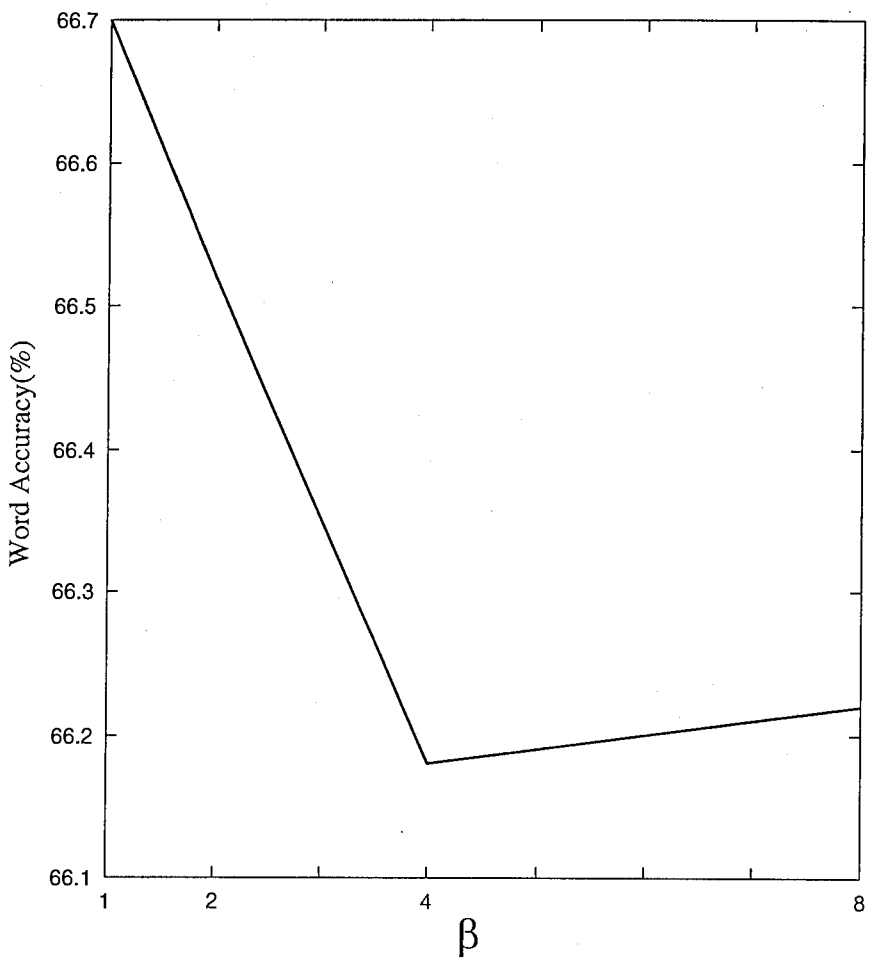

図 8: 発音尤度の重み β を変えたときの認識率

付録 D　辞書生成用プログラムのパッケージの内容

辞書生成用プログラムのパッケージの内容は以下のようになっている．(場所 ~xtyoshim/genprn)

ファイル		内容
Makefile		Makefile
README		プログラム, スクリプトの使い方をまとめたもの
genlex.c		辞書作成用プログラム
genprn.c		発声変形用プログラム
program_test.pl		プログラムテスト用スクリプト
mlp1.35/		NN の学習プログラムパッケージ
script/	mknif.pl	音素タイプライタの結果から NN のトレーニング用データを作成する
	splex.pl	尤度付き発音辞書から，ID を変えた確率版 LM, Classfile，MergeList を出力する
	modprnprob.pl	尤度付き発音辞書の尤度に，発音列の音素数に従い期待値を掛ける
	sumlex.pl	pause を考慮した発音辞書を作る
	sumplex.pl	pause を考慮した尤度付き発音辞書を作る
	mkngram.csh	言語モデルを作成する
sample/	Bigram	単語バイグラムのデータ
	LEX.W.exp	LEX.W を展開したもの
	LEX.W.new	pause の付かない従来辞書
	LEX.W.new.pau	pause の付いた従来辞書
	LEX.W.new.pau.exp	LEX.W.new.pau を展開したもの
	train.mlp.s500	alpha のマシンでトレーニングした NN
test_sample/	NN	テスト用 NN
	LEXICON	テスト用 LEXICON
	BIGRAM	テスト用 BIGRAM
	NEWLEXICON	NN,LEXICON,BIGRAM を用いて作成した LEXICON

付録 E　パッケージ内のプログラム，スクリプトの使い方

E.1　genprn の用途と使い方

NN を使って，標準入力から読み込んだ発音系列を変化させ出力する．

```
Usage    : genprn [option] mlpfile

  mlpfile - DmlpTrain で学習した NN   (train.mlp.s???)

Options :
  -s     : 音素毎のスコアを出す．(default)
  -n     : N(single)
  -m     : N(multi)
  -l     : 発音列を尤度付で出す．
  -N n   : n 個の発音列を出す．

Example :
% genprn train.mlp.s500
a i u e o
a
i
u 1.0000/       ! 0.2191/
e
o
% genprn -n train.mlp.s500
a i u e o
a i u e o
% genprn -m train.mlp.s500
a i u e o
a i u e o
a i e o
% genprn -l -m -N 8 train.mlp.s500
k a m o g a w a r j o k a ng
1.000000 : k a m o g a w a r j o k a ng
0.379221 : k a m o a w a r j o k a ng
0.362647 : k a m o g a w a r j o p a ng
0.356487 : k a m o g o w a r j o k a ng
0.272306 : k a m o g w a r j o k a ng
0.249761 : k a m o g a a r j o k a ng
0.242694 : k a m o g a o a r j o k a ng
0.221625 : k a m o g a w a j o k a ng

Bug      :
  プログラムを実行するマシンは，NN をトレーニングした
  マシンとバイトオーダ，ビット数が同じでなければならない．
```

動作確認:

```
  OSF1 V3.2 alpha(gcc2.7.2.2)
  OSF1 V4.0 alpha(gcc2.7.2.2)
```

E.2　genlex の用途と使い方

NN を使って，単語辞書の発音系列を変化させ出力する．

```
Usage    : genlex [option] mlpfile lexicon bigram > newlexicn
  mlpfile - DmlpTrain で学習した NN   (train.mlp.s???)
  lexicon - 発音変形させる単語辞書  (LEX.W.exp など)
  bigram  - Bigram (sample/Bigram)

Options :
  -c f   : 発音列を変形させて，元の発音列と違った場合，
           発音尤度を f として，元の発音列を加える．
  -C     : 発音列を変形させて，元の発音列と違った場合，
           発音尤度を 0.01 として，元の発音列を加える．
  -d     : 尻 2 音素のみを変える．
  -l     : 頭 2 音素のみを変える．
  -i     : 両端 2 音素を変えない．(bigram を読まない)
  -m i   : 変化させる発音列の最小音素数を i (i > 3) 個にする．
           (音素数が i 個以上の発音列は変化させない．)
  -p     : 発音尤度付き辞書を出力する．
  -t f   : Threshold を f にする．(default 1)
  -w f   : 発音尤度 p を pow(p,f) にする．(default 1)
  -z     : 発音尤度の正規化を行なわない．
  -f     : pause を考慮した辞書作成を行なう．

Bug      :
  プログラムを実行するマシンは，NN をトレーニングした
  マシンとバイトオーダ，ビット数が同じでなければならない．
```

動作確認:

```
  OSF1 V3.2 alpha(gcc2.7.2.2)
  OSF1 V4.0 alpha(gcc2.7.2.2)
```

E.3　script/splex.pl の用途と使い方

尤度付き発音辞書から，ID を変えた確率版言語モデル，Classfile，MergeList を出力する．

```
Usage    : splex.pl (入力) 尤度付き発音辞書 (出力)ID を変えた発音辞書          \
                    (出力)MergeList                                          \
                    (入力) 確率版言語モデル (出力)ID を変えた確率版言語モデル \
                    (入力)Classfile          (出力)ID を変えた Classfile
```

E.4　script/mkngram.csh の用途と使い方

ascii の LM を binary にする．

```
Usage    : mkngram.csh (入力)Lexicon (入力)Classfile (入力)ascii 版言語モデル \
                       (出力)binary 版言語モデル
```

E.5 script/sumlex.pl の用途と使い方

pause を考慮した発音辞書を作る.

```
Usage   : sumlex.pl Lexicon1 Lexicon2 > Lexicon
  Lexicon1 : pause 無しの従来辞書で言語統計量を用いて発声変形
  Lexicon2 : pause 無しの従来辞書で言語統計量を用いず発声変形
  Lexicon  : merge(Lexicon1+Lexicon2)+add_pause(Lexicon2)
```

E.6 script/sumplex.pl の用途と使い方

尤度つきの pause を考慮した発音辞書を作る.

```
Usage   : sumplex.pl Lexicon1 Lexicon2 > Lexicon
  Lexicon1 : pause 無しの従来辞書で言語統計量を用いて発声変形
  Lexicon2 : pause 無しの従来辞書で言語統計量を用いず発声変形
  Lexicon  : merge(Lexicon1+Lexicon2)+add_pause(Lexicon2)
```

E.7 script/modprnprob.pl の用途と使い方

尤度付発音辞書の各発音尤度に，発音列の音素数に従い信頼度を掛ける.

```
Usage   : modprnprob.pl [options] LEX.W.new (入力) 尤度付き発音辞書 \
                        > 修正尤度付き発音辞書

Options :
  -ht   : 入力の尤度付き発音辞書が言語統計量を用いて作成されている場合
          1 音素隣り 0.8, 2 音素隣り 0.9
```

E.8 script/mknif.pl の用途と使い方

音素タイプライター認識の結果を用いて,
発音ネットワークのトレーニング用データを作成する.

```
Usage   : mknif.pl ResultFile ...
```

付録 F　発音辞書自動生成の手順

発音ネットワーク作成から，発音辞書自動生成までの手順を以下に示す．

F.1　発音ネットワークを作成する

1. 音素タイプライタ認識を行う．

2. 音素タイプライタ認識の結果 (付録 G 参照) から，ニューラルネット学習用のデータを作成する．
   % mknif.pl TAC70201.A.res ...
   mknif.pl から出力されるもの : datafile.in(NN の入力) datafile.out(NN の出力)

3. NN をトレーニングする準備をする．

```
% mkdir data
% cat > header.in                  # 入力データのヘッダを作る
NN_ascii_data                      # ヘッダ
118564                             # 入力データの数
130                                # 入力層の数
^D
% cat > header.out                 # 出力データのヘッダを作る
NN_ascii_data                      # ヘッダ
118564                             # 出力データの数
53                                 # 出力層の数
^D
% cat header.in datafile.in   > data/datafile.in
% cat header.out datafile.out > data/datafile.out
% echo "data/datafile.in"  > train.list_in
% echo "data/datafile.out" > train.list_out
% cat > train.mlp_def
LAYERS  2                          # 層の数
IN      130                        # 入力層の数
HIDDEN1 100                        # 中間層の数
OUT     53                         # 出力層の数
^D
```

4. NN をトレーニングする．
   % Dmlptrain -n_cycles 1000 -r_delta_start 0.01 -seed 0 -rand_low -0.01 -rand_up 0.01 train

F.2　発声変形させる辞書を作成する

- 従来辞書の場合

  1. LEX.W のカタカナ読みフィールドから音素列を生成する．
     アリガトウ → a r i g a t o u

  2. 辞書を作成する．

```
5 [UTT-START] -  # UTT-START
6 [UTT-END] -  # UTT-END
10000 [いらっしゃいませ] i r a q sh j a i m a s e  # いらっしゃいませ|
イラッシャイマセ|いらっしゃいませ|感動詞||||
10005 [あ] a  # あ|ア|あ|間投詞||||
10007 [一] h i t o  # 一|ヒト|一|数詞||||
10008 [つ] ts u  # つ|ツ|つ|接尾辞||||
```

...
...

- エキスパート辞書の場合

1. LEX.W の発音の正規表現を展開した辞書を作る.

```
10009 [部屋] {h|b} e j a {|-}  # 部屋 | ヘヤ | 部屋 | 普通名詞 ||||
```

⇓

```
10009 [部屋] h e j a  # 部屋 | ヘヤ | 部屋 | 普通名詞 ||||
10009 [部屋] h e j a -  # 部屋 | ヘヤ | 部屋 | 普通名詞 ||||
10009 [部屋] b e j a  # 部屋 | ヘヤ | 部屋 | 普通名詞 ||||
10009 [部屋] b e j a -  # 部屋 | ヘヤ | 部屋 | 普通名詞 ||||
```

F.3　発音ネットワークと辞書を入力して発声変形する

(F.3.1)　従来辞書の場合

- 発音尤度と言語統計量を用いない場合.

```
% genlex -i -t 0.4 -C        \ # 言語統計量を使わない, threshold 0.4, N(multi)+C
    sample/train.mlp.s500    \ # 発音ネットワーク
    sample/LEX.W.new.pau.exp \ # 辞書
    > LEX.W.new.pau.exp.NmC.s500.t0.4 # 発音辞書
```

- 発音尤度のみ用いる場合.

```
% genlex -i -p -C -t 0.4     \ # 言語統計量を使わない, 尤度付き, threshold 0.4, N(multi)+C
    sample/train.mlp.s500    \ # (入力) 発音ネットワーク
    sample/LEX.W.new.pau.exp \ # (入力) 辞書
    > LEX.W.new.pau.exp.NmC.s500.t0.4.scr # (出力) 尤度付き発音辞書
% script/modprnprob.pl                    \
    sample/LEX.W.new                      \ # (入力) 標準発音列の音素数を見るための辞書
    LEX.W.new.pau.exp.NmC.s500.t0.4.scr \ # (入力) 言語統計量を用いず作成された尤度付き発
音辞書
    > LEX.W.new.pau.exp.penalty.NmC.s500.t0.4.scr # (出力) 発音尤度に信頼度を掛けた発音辞
書
% script/splex.pl                                  \
    LEX.W.new.pau.exp.penarty.NmC.s500.t0.4.scr \ # (入力) 発音尤度に信頼度を掛けた発音辞
書
    LEX.W.new.pau.exp.penarty.NmC.s500.t0.4     \ # (出力) ID を書き換えた発音辞書
    MergeList.new.pau.penarty.NmC.s500.t0.4     \ # (出力) MergeList
    vngram.travel.prob.1000                     \ # (入力) 確率版言語モデル
    Gram.new.pau.penalty.NmC.s500.t0.4.prob     \ # (出力) 確率に発音尤度を掛けた言語モデ
ル
    vnclass.travel.1000                         \ # (入力) ClassFile
    Class.new.pau.penalty.NmC.s500.t0.4           # (出力) ID を書き換えた ClassFile
% script/mkngram.csh                           \
    LEX.W.new.pau.penalty.NmC.s500.t0.4         \ # (入力) ID を書き換えた発音辞書
    Class.new.pau.penalty.NmC.s500.t0.4         \ # (入力) ID を書き換えた ClassFile
    Gram.new.pau.penalty.NmC.s500.t0.4.prob     \ # (入力) 確率に発音尤度を掛けた言語モデ
ル
    Gram.new.pau.penalty.NmC.s500.t0.4.prob.bin   # (出力) binary の言語モデル
```

- 言語統計量のみ用いる場合.

```
% genlex -C -t 0.4           \  # threshold 0.4, N(multi)+C
    sample/train.mlp.s500    \  # (入力) 発音ネットワーク
    sample/LEX.W.new.pau.exp \  # (入力) 辞書
    sample/Bigram            \  # (入力) 単語バイグラム
    > LEX.W.etc.NmC.s500.t0.4.1 # (出力) 言語統計量を用いて作成された発音辞書
% genlex -i -C -t 0.4     \      # 言語統計量を用いない, threshold 0.4, N(multi)+C
    sample/train.mlp.s500    \  # (入力) 発音ネットワーク
    sample/LEX.W.new.pau.exp \  # (入力) 辞書
    > LEX.W.etc.NmC.s500.t0.4.2 # (出力) 言語統計量を用いず作成された発音辞書
% script/sumlex.pl            \
    LEX.W.etc.NmC.s500.t0.4.1 \ # (入力) 言語統計量を用いて作成された発音辞書
```

```
    LEX.W.etc.NmC.s500.t0.4.2 \ # (入力) 言語統計量を用いず作成された発音辞書
    > LEX.W.etc.NmC.s500.t0.4   # (出力)2 つの発音辞書をマージした発音辞書
```

- 発音尤度と言語統計量ともに用いる場合.

```
% genlex -p -C -t 0.4     \          # 尤度付き, threshold 0.4, N(multi)+C
    sample/train.mlp.s500 \          # (入力) 発音ネットワーク
    sample/LEX.W.new      \          # (入力) 辞書
    sample/Bigram         \          # (入力) 単語バイグラム
    > LEX.W.etc.NmC.s500.t0.4.scr.1 # (出力) 言語統計量を用いて作成された尤度つき発音辞書
% script/modprnprob.pl -ht        \ # 言語統計量を用いている
    sample/LEX.W.new              \ # (入力) 標準発音列の音素数を見るための辞書
    LEX.W.etc.NmC.s500.t0.4.scr.1 \ # (入力) 言語統計量を用いて作成された尤度つき発音辞書
    > LEX.W.etc.penalty.NmC.s500.t0.4.scr.1 # (出力) 発音尤度に信頼度を掛けた発音辞書 1
% genlex -p -i -C -t 0.4    \        # 尤度付き, threshold 0.4, N(multi)+C
    sample/train.mlp.s500   \        # (入力) 発音ネットワーク
    sample/LEX.W.new        \        # (入力) 辞書
    > LEX.W.etc.NmC.s500.t0.4.scr.2 # (出力) 言語統計量を用いず作成された尤度付き発音辞書
% script/modprnprob.pl            \ # 言語統計量を用いていない
    sample/LEX.W.new              \ # (入力) 標準発音列の音素数を見るための辞書
    LEX.W.etc.NmC.s500.t0.4.scr.2 \ # (入力) 言語統計量を用いず作成された尤度付き発音辞書
    > LEX.W.etc.penalty.NmC.s500.t0.4.scr.2 # (出力) 発音尤度に信頼度を掛けた発音辞書 2
% script/sumplex.pl                        \
    LEX.W.etc.penalty.NmC.s500.t0.4.scr.1 \ # (入力) 発音尤度に信頼度を掛けた発音辞書 1
    LEX.W.etc.penalty.NmC.s500.t0.4.scr.2 \ # (入力) 発音尤度に信頼度を掛けた発音辞書 2
    > LEX.W.etc.penalty.NmC.s500.t0.4.scr   # (出力)2 つの発音辞書をマージした尤度付き発音
辞書
% script/splex.pl                          \
    LEX.W.etc.penalty.NmC.s500.t0.4.scr   \ # (入力)2 つの発音辞書をマージした尤度付き発音
辞書
    LEX.W.etc.penalty.NmC.s500.t0.4       \ # (出力)ID を書き換えた発音辞書
    MergeList.etc.penalty.NmC.s500.t0.4   \ # (出力)MergeList
    vngram.travel.prob.1000               \ # (入力) 確率版言語モデル
    Gram.etc.penalty.NmC.s500.t0.4.prob   \ # (出力) 確率に発音尤度を掛けた言語モデル
    vnclass.travel.1000                   \ # (入力)ClassFile
    Class.etc.penalty.NmC.s500.t0.4         # (出力)ID を書き換えた ClassFile
% script/mkngram.csh                       \
    LEX.W.etc.penalty.NmC.s500.t0.4       \ # (入力)ID を書き換えた発音辞書
    Class.etc.penalty.NmC.s500.t0.4       \ # (入力)ID を書き換えた ClassFile
    Gram.etc.penalty.NmC.s500.t0.4.prob   \ # (入力) 確率に発音尤度を掛けた言語モデル
    Gram.etc.penalty.NmC.s500.t0.4.prob.bin # (出力)binary の言語モデル
```

(F.3.2) エキスパート辞書の場合

- 発音尤度と言語統計量を用いない場合. (threshold 0.4)

```
% genlex -i -t 0.4 -C     \ # 言語統計量を使わない, threshold 0.4, N(multi)+C
    sample/train.mlp.s500 \ # 発音ネットワーク
    sample/LEX.W.exp      \ # 辞書
    > LEX.W.exp.NmC.s500.t0.4 # 発音辞書
```

- 発音尤度のみ用いる場合.

```
% genlex -i -p -C -t 0.4  \ # 言語統計量を使わない，尤度付き，threshold 0.4, N(multi)+C
    sample/train.mlp.s500 \ # (入力) 発音ネットワーク
    sample/LEX.W.exp      \ # (入力) 辞書
    > LEX.W.exp.NmC.s500.t0.4.scr # (出力) 尤度付き発音辞書
% script/modprnprob.pl                    \
    sample/LEX.W.new                     \ # (入力) 標準発音列の音素数を見るための辞書
    LEX.W.exp.NmC.s500.t0.4.scr          \ # (入力) 言語統計量を用いず作成された尤度付き発
音辞書
    > LEX.W.exp.penalty.NmC.s500.t0.4.scr # (出力) 発音尤度に信頼度を掛けた発音辞書
% script/splex.pl                         \
    LEX.W.exp.penalty.NmC.s500.t0.4.scr \ # (入力) 発音尤度に信頼度を掛けた発音辞書
    LEX.W.exp.penalty.NmC.s500.t0.4      \ # (出力) ID を書き換えた発音辞書
    MergeList.penalty.NmC.s500.t0.4      \ # (出力) MergeList
    vngram.travel.prob.1000              \ # (入力) 確率版言語モデル
    Gram.penalty.NmC.s500.t0.4           \ # (出力) 確率に発音尤度を掛けた言語モデル
    vnclass.travel.1000                  \ # (入力) ClassFile
    Class.penalty.NmC.s500.t0.4            # (出力) ID を書き換えた ClassFile
% script/mkngram.csh                      \
    LEX.W.exp.penalty.NmC.s500.t0.4      \ # (入力) ID を書き換えた発音辞書
    Class.penalty.NmC.s500.t0.4          \ # (入力) ID を書き換えた ClassFile
    Gram.penalty.NmC.s500.t0.4           \ # (入力) 確率に発音尤度を掛けた言語モデル
    Gram.penalty.NmC.s500.t0.4.bin         # (出力) binary の言語モデル
```

- 言語統計量のみ用いる場合.

```
% genlex -f -C -t 0.4      \  # pause を考慮する，threshold 0.4, N(multi)+C
    sample/train.mlp.s500 \  # (入力) 発音ネットワーク
    sample/LEX.W.exp      \  # (入力) 辞書
    sample/Bigram         \  # (入力) 単語バイグラム
    > LEX.W.exp.ignore_pau.NmC.s500.t0.4 # (出力) 言語統計量を用いて作成された発音辞書
```

- 発音尤度と言語統計量ともに用いる場合.

```
% genlex -p -f -C -t 0.4  \  # 尤度付き，threshold 0.4, N(multi)+C
    sample/train.mlp.s500 \  # (入力) 発音ネットワーク
    sample/LEX.W.exp      \  # (入力) 辞書
    sample/Bigram         \  # (入力) 単語バイグラム
    > LEX.W.exp.ignore_pau.NmC.s500.t0.4.scr # (出力) 言語統計量を用いて作成された尤度つき
発音辞書
% script/modprnprob.pl -ht                \ # 言語統計量を用いている
    sample/LEX.W.new                     \ # (入力) 標準発音列の音素数を見るための辞書
    LEX.W.exp.ignore_pau.NmC.s500.t0.4.scr \ # (入力) 言語統計量を用いて作成された尤度つき
発音辞書
    > LEX.W.exp.ignore_pau.penalty.NmC.s500.t0.4.scr # (出力) 発音尤度に信頼度を掛けた発音
辞書
% script/splex.pl                                  \
    LEX.W.exp.ignore_pau.penalty.NmC.s500.t0.4.scr \ # (入力) 発音尤度に信頼度を掛けた発音
辞書
    LEX.W.exp.ignore_pau.penalty.NmC.s500.t0.4     \ # (出力)ID を書き換えた発音辞書
    MergeList.ignore_pau.penalty.NmC.s500.t0.4     \ # (出力)MergeList
    vngram.travel.prob.1000                        \ # (入力) 確率版言語モデル
```

```
    Gram.ignore_pau.penalty.NmC.s500.t0.4          \ # (出力) 確率に発音尤度を掛けた言語モ
デル
    vnclass.travel.1000                            \ # (入力)ClassFile
    Class.ignore_pau.penalty.NmC.s500.t0.4           # (出力)ID を書き換えた ClassFile
% script/mkngram.csh                               \
    LEX.W.exp.ignore_pau.penalty.NmC.s500.t0.4     \ # (入力)ID を書き換えた発音辞書
    Class.ignore_pau.penalty.NmC.s500.t0.4         \ # (入力)ID を書き換えた ClassFile
    Gram.ignore_pau.penalty.NmC.s500.t0.4          \ # (入力) 確率に発音尤度を掛けた言語モ
デル
    Gram.ignore_pau.penalty.NmC.s500.t0.4.bin        # (出力)binary の言語モデル
```

F.4 認識実験を行う

発音尤度を用いている場合は sample/splex.pl, sample/mkngram.csh で作成した言語モデル, MergeList を, ATRlattice, ATRresult で使うように設定する.

- 発音尤度を用いていない場合

```
% cat 音声パラメータ | ATRlattice -UTT_START=''5'' -UTT_END=''6''  \
    -lexicon= 発音辞書 -amname= 音響モデル -ngram=Class-2, 言語モデル \
    -config=config.recog >! LatticeFile
% ATRresult < LatticeFile -config=config.result -answer=AnswerFile
```

- 発音尤度を用いている場合

```
% cat 音声パラメータ | ATRlattice -UTT_START=''5aaa'' -UTT_END=''6aaa'' \
    -lexicon=ID を書き換えた発音辞書 -amname= 音響モデル                  \
    -ngram=Class-2, 作成した言語モデル -config=config.recog >! LatticeFile
% ATRresult < LatticeFile -config=config.result -merge_list=MergeList  \
    -answer=AnswerFile
```

付録 G　音素タイプライタ認識の結果の例

```
VERSION=ReleaseATRLATTICE 04r07a OSF1
frameshift=0.010000
base=1.000100
lmname=Lexicon(/dept1/work1/V1/model/LEX.P),Ngram(/dept1/work1/V1/model/LM.
P)
lmscale=8.000000

# Answer  : 5 s u m i m a s e ng i ch i n i ch i - t a i z a i o e ng ch j
o o sh i t a i ng d e s u k e r e d o m o - 6
# Best list
# Dp_list : 5 s u $D i m a s e ng $I $I i ch i n i ch i $I $S t a $S z a $S
 $S $S ng ch j o o sh i t a i ng d $S s u k e r e d o m o - 6
# Score = 10.000000
# phone = 49, ins = 3, del = 1, sub = 6
# Network
# Dp_list : 5 s u m i m a s e ng $I i ch i n i ch i $I $S t a i z a $S $S $
S ng ch j o o sh i t a i ng d $S s u k e r e d o m o - 6
# Score = 7.000000
# phone = 49, ins = 2, del = 0, sub = 5
# Dp_list : 5 s u m i m a s e ng i $I $I ch i n i $I ch i $I $I $S t a i z
a $I $I $I i $D $D ng ch j o o sh i t a i ng d $S s u k e r e $I d o m o -
6
# Score = 4.000000
# phone = 49, ins = 9, del = 2, sub = 2
UTTERANCE=1
amname=[/usr/tmp/atra55.2024.252.bin]
utime=5.200000
abstime=0.000000
cputime=1.332000
NBEST=1

ORDER=1 WORDS=5/s/u/i/m/a/s/e/ng/i/-/i/ch/i/n/i/ch/i/i/q/t/a/e/z/a/w/a/i/ng
/ch/j/o/o/sh/i/t/a/i/ng/d/a/s/u/k/e/r/e/d/o/m/o/-/6 wordids=5/s/u/i/m/a/s/e
/ng/i/-/i/ch/i/n/i/ch/i/i/q/t/a/e/z/a/w/a/i/ng/ch/j/o/o/sh/i/t/a/i/ng/d/a/s
/u/k/e/r/e/d/o/m/o/-/6 vars=1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1
/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1/1 divs=-/s/u/i/m/a
/s/e/ng/i/-/i/ch/i/n/i/ch/i/i/q/t/a/e/z/a/w/a/i/ng/ch/j/o/o/sh/i/t/a/i/ng/d
/a/s/u/k/e/r/e/d/o/m/o/-/- times=0.000000/0.570000/0.730000/0.800000/0.9100
00/0.980000/1.050000/1.120000/1.190000/1.260000/1.350000/1.380000/1.470000/
1.580000/1.640000/1.690000/1.780000/1.900000/2.080000/2.150000/2.430000/2.4
90000/2.580000/2.630000/2.710000/2.850000/2.890000/2.950000/3.020000/3.1300
00/3.190000/3.230000/3.280000/3.330000/3.390000/3.430000/3.470000/3.540000/
3.600000/3.670000/3.720000/3.810000/3.880000/3.950000/4.010000/4.060000/4.1
30000/4.200000/4.250000/4.320000/4.410000/4.630000/5.200000 score=28759381.
000000 acoustic=28759381.000000 ngram=0.000000
...
...
省略
```

付録 H　単語バイグラムの例

単語バイグラムのフォーマットは, 頻度 単語 ID 後続単語 ID の様になっている.

```
2   10015 11468
4   10288 10418
2   5 15790
1   10070 10438
1   13036 10215
2   11420 10320
8   10368 18836
2   10806 13180
1   10098 11985
1   16108 10413
1   10074 10921
2   15707 10010
1   13739 10183
7   10049 10923
3   11655 18306
1   12870 14304
1   10060 11989
2   15708 10010
1   10289 10418
2   10257 14489
1   14158 10154
1   10049 16597
1   10356 18839
2   10056 14955
1   11819 14368
2   10017 11468
1   15709 10010
1   16786 10341
1   10010 11996
...
...
```

省略