音声言語情報処理 28−6
(1999. 10. 29)

N-best 候補からの言語重みと挿入ペナルティの最適化に関する検討

伊藤 彰則，好田 正紀

山形大学 工学部

大語彙連続音声認識システムには，最適化すべき多くのパラメータがある．本稿では，その中で言語重みと挿入ペナルティの設定について考察する．まず，実際に言語重みと挿入ペナルティの値を変えながら認識実験を行った結果から，三つの重要な観察事実を得た．一つ目は，これらのパラメータの最適値が，最適化に用いたテストセットに依存するという点である．二つ目は，これらのパラメータと単語エラー率の関係は複雑で，大域的な最適解を得るためには，パラメータ空間を全探索しなければならないという点である．三つ目は，ある程度最適な領域における単語エラー率の変動は2%程度という点である．これらの事実を踏まえて，パラメータ最適化のための新しい手法を三つ提案する．最初に，n-best 候補のリスコアリングに基づいてパラメータを最適化する場合に，その n-best 候補を予備選択する方法を提案する．この方法を使えば，最適化結果をまったく変えずに，n-best 候補の数を大幅に削減することができる．二つ目に，より頑健なパラメータの設定値を探す方法を提案する．この方法は，パラメータ最適化のためのテストセットをいくつかに分割し，あるパラメータ設定における単語エラー率の平均と分散を考慮するという方法であり，bigram 言語モデルを用いた場合には，ある程度の頑健性の改善が得られている．最後に，近隣サーチに基づいて，準最適なパラメータ設定を高速に探索する手法を提案する．

Fast and Robust Optimization of Language Model Weight and Insertion Penalty from N-best Candidates

Akinori Ito and Masaki Kohda

Faculty of Engineering, Yamagata University

An LVCSR system has many parameters to be optimized. In this paper, we investigate several issues about language model weight and word insertion penalty. From recognition results obtained by changing these parameters, we found three important observations. The first one was that the optimum point of these parameter values depended to the test set for the optimization. The second one was that the parameter space had many local optimum, which meant that one had to try all points in the parameter space to find the global optimum point. The third one was that the potential increment of WER in suboptimum region of the parameter space was about 2%. Based on these observations, We propose three new methods to optimize language model weight and insertion penalty. Firstly, a new method is proposed to preselect n-best candidates for n-best rescoring based parameter optimization. Secondly, a method to choose robust parameter setting is proposed. This method splits a development test set into several sets. Accoding to the optimization results for each set, This method choosed the optimum point by considering the average of WER as well as its variances. Finally, a method to find sub-optimum parameter setting is proposed. This optimization is based on neighborhood search, and it finds a parameter setting rapidly.

1 はじめに

ほとんどの大語彙連続音声認識システムには，システムを調整するための多くのパラメータがある．例えば，言語重み，挿入ペナルティ，ビーム幅などである．これらのパラメータは，開発用のテストセットに対して単語エラーが最も低くなるように設定されるのが普通である．これらのパラメータを効率良く最適化するための手法は現在のところ知られていないので，最適なパラメータ設定のためには，原理的にはこれらのパラメータの全ての組みあわせについて認識実験を

しなければならない[1]．しかし，それには莫大な時間がかかる．それだけでなく，開発用のテストセットについて最適なパラメータ設定が，その他のデータに対しても最適である保証はない．

多くの場合，ある音声入力に対して最適であるようなパラメータ設定が，その他の入力に対しても最適であるとは限らない．これら認識パラメータの設定が重要であることは明らかだが，そのパラメータ設定自体の頑健性に注目した研究はこれまでほとんどなかった．

本稿では，言語重みと挿入ペナルティの最適化にまつわるいくつかの問題点を明らかにし，その一部を解決しようとしている．

2. では，n-best 候補からのパラメータ最適化を定式化し，実際に最適化をしたときの結果について考察する．ここでの観察の結果から，3 つの結論が導かれる．最初の結論は，パラメータが開発用テストセットに依存するということである．これは，入力音声によらず最適であるようなパラメータ設定は不可能だということを意味する．第 2 の結論は，言語重みと挿入ペナルティのパラメータ空間上には無数の局所最適点があるため，大域的に良い解を効率的に探索するのは難しそうだということである．第 3 の結論は，上記の問題にもかかわらず，「ほぼ」良い解を見つけることはできそうだということである．

3. では，パラメータの最適化を高速にするための n-best 候補の予備選択法を提案する．これは非常に簡単な方法だが，n-best 候補の 87% を削減することができ，最適化を 8 倍高速化できた．

4. では，パラメータ設定の頑健性について議論する．その後，より頑健なパラメータ設定を探すための方法を提案する．基本的なアイデアは，開発用テストセットを複数のサブセットに分割し，それぞれのサブセットに対する最適パラメータ設定から，全体のパラメータ設定を決定するというものである．

5. では，高速に準最適なパラメータ設定を探すアルゴリズムを提案する．この方法は，近隣サーチ (neiborhood search) に基いて格子状のパラメータ空間を探索する手法である．実験により，この手法の性能を評価する．

2 N-best 候補によるパラメータ最適化

2.1 言語モデル重みと挿入ペナルティ

音声認識システムが認識候補を生成すると，その候補はそのスコアに基いて評価される．通常のシステムでは，一つの候補は音響スコア x_A と言語スコア x_L を持っている．x を音響信号，w を単語列とするとき，これらのスコアはそれぞれ $\log P(x|w)$ と $\log P(w)$ であることが多い．しかし，これらのスコアは対数確率でないこともある．実際，連続分布型 HMM から出力されるスコアは確率密度であって確率ではなく，1 を越えることもあり得る．

複数の認識候補を一次元の量で比較するため，音響スコアと言語スコアから総合スコアを計算する．このとき，候補の単語数 n も考慮に入れられることが多い．最も一般的なのは，これらの線型結合である．

$$x = a_1 x_A + a_2 x_L + a_3 n \tag{1}$$

候補間の良さの比較のためには，スコアの相対的な大きさだけが意味を持つ．そこで，a_1 が正だと仮定して，次のような式が良く用いられる．

$$\begin{aligned} x &= x_A + \frac{a_2}{a_1} x_L + \frac{a_3}{a_1} n & (2)\\ &= x_A + W_L x_L + P_I n & (3) \end{aligned}$$

W_L と P_I は，それぞれ言語重みと挿入ペナルティと呼ばれる．パラメータの最適化過程では，単語エラー率が最小となるように W_L と P_I を選ぶ．

言語重みと挿入ペナルティは，言語スコアの分布と音響スコアの分布の違いを補償する働きを持つ．これは，$x_L = \sum_i \log P(w_i|w_1^{i-1})$ とした場合の，次のような式からも明らかであろう．

$$\begin{aligned} x &= x_A + W_L x_L + P_I n & (4)\\ &= x_A + W_L \left\{ \sum_i \left(\log P(w_i|w_1^{i-1}) + \frac{P_I}{W_L} \right) \right\} \\ &= x_A + \frac{\sum_i (\log P(w_i|w_1^{i-1}) - \mu)}{\sigma} \end{aligned}$$

ただし $\sigma = W_L^{-1}, \mu = -P_I/W_L$ である．問題は両スコアの分布をどのように調整するかであるが，これに一般的な解を求めるのは難しい．これについては後述する．

2.2 N-best 候補を用いた最適化

言語重みと挿入ペナルティを最適化するには，(W_L, P_I) の全ての組みあわせについて認識実験をすればよい．しかし，全ての組みあわせについて「本当の」認識実験をすると，非常に時間がかかる．そこで，最適化を高速にするため，n-best 候補に基づく実験が行われることがある．まず，あるパラメータ設定の元で，各発話に対して n-best 候補を計算する．その後，n-best 候補のリスコアリングによってパラメータの最適化を行うのである．

この n-best に基づくパラメータの最適化を定式化する．まず，次の記号を定義する．

M	発話数
N	それぞれの発話に対する候補数
$x(i,j)$	i 番目の発話の j 番目の候補の総合スコア
$x_A(i,j)$	i 番目の発話の j 番目の候補の音響スコア
$x_L(i,j)$	i 番目の発話の j 番目の候補の言語スコア
$n(i,j)$	i 番目の発話の j 番目の候補の単語数
$E(i,j)$	i 番目の発話の j 番目の候補の誤認識単語数

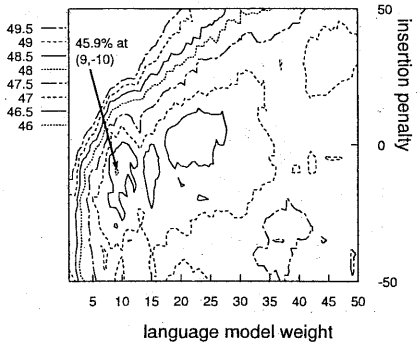

図 1: 偶数データに対する最適化結果

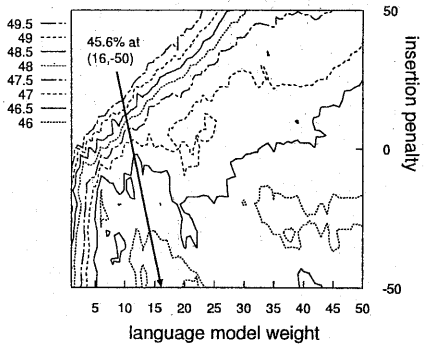

図 2: 奇数データに対する最適化結果

W_L と P_I は次のようにして最適化される.

$$\tilde{n}_i(W_L, P_I) = \mathrm{argmax}_j \{x_A(i,j) + W_L x_L(i,j) + P_I n(i,j)\} \quad (5)$$

$$\tilde{W}_L, \tilde{P}_I = \underset{W_L, P_I}{\mathrm{argmin}} \sum_{i=1}^{M} E(i, \tilde{n}_i(W_L, P_I)) \quad (6)$$

$\tilde{n}_i(W_L, P_I)$ は, パラメータ設定 (W_L, P_I) の元での, i 番目の発話に対する最適な候補の番号である.

2.3 認識パラメータの頑健性

既に述べた通り, パラメータの最適化は, 開発テストセットに対する認識結果に基いて行われる. この時点で評価セットは未知であるため, 開発テストセットに基づくパラメータが, 評価セットに対してどの程度頑健であるかは明らかではない. そこで, パラメータの頑健さを調べる実験を行った.

実験に用いたデータは, Switchboard corpus の中の 684 発話から, BBN のデコーダによって各発話に対して最大 100best 候補を計算したものである. 言語モデルには bigram を使用している.

最初に, n-best の結果を 2 つのサブセットに分割する. この実験では, 偶数番目の発話 (337 発話, 3184 単語) と奇数番目の発話 (347 発話, 3192 単語) に分割した. 次に, それぞれのサブセットを使って, パラメータを最適化する. このとき, $W_L = (1, 2, \ldots, 50)$ と $P_I = (-50, -45, -40, \ldots, 45, 50)$ の全ての組みあわせについてリスコアリングを行った. このときの結果を, 図 1と図 2に示す. どちらの場合においても, 言語重みが低く, 挿入ペナルティが高い領域では単語エラーが高くなっている. しかし, それぞれのサブセットに対する最適点は全く異っている.

これらの結果はどの程度違うのだろうか. 次の表は, 片方のサブセットで最適化したパラメータを使って, もう片方のサブセットをリスコアリングしたときの結果を示したものである. 表中の total と書かれた行は, 全体のテストセットに対する最適化結果ではなく, それぞれの最適化結果の合計である. この結果から, 他のセットを使って最適化をすることで, 単語エラー率が 1 ～ 2 ポイント上昇することがわかる. もしこの程度の単語エラー率が問題になる場合には, パラメータの設定方法にもっと気を配った方が良いということになる. 4. では, より頑健なパラメータ設定法について述べる.

表 1: 偶数および奇数セットの最適化結果

data set	当該セットによる最適化結果			
	#sub	#ins	#del	WER%
EVEN	859	244	359	45.9
ODD	842	149	466	45.6
total	1701	393	825	45.8
data set	違うセットによる最適化結果			
	#sub	#ins	#del	WER%
EVEN	861	189	448	47.0
ODD	867	234	396	46.9
total	1728	423	844	46.9

上記の最適化結果からわかるもう一つの特徴は, (W_L, P_I) の空間に多くの局所最適点があるということである. これはすなわち, 効率的に大域最適解を求めるのが難しいということを意味する. しかし, 1 ～ 2 ポイント程度の単語エラー率の上昇が問題でなければ, 大域的に最適な単語エラー率＋ 1% 程度のエラー率を与えるパラメータを探すことはそれほど困難ではない. 表 2は, ここで調査した全てのパラメータ空間

表 2: パラメータの局所最適点と単語誤り率

EVEN				ODD			
順位	W_L	P_I	WER	順位	W_L	P_I	WER
1	9	-10	45.9	1	16	-50	45.6
2	21	5	46.1	2	15	-35	45.7
3	19	0	46.2	3	12	-35	45.7
4	23	-5	46.2	4	19	-45	45.7
5	9	-20	46.2	5	36	-25	45.8
6	24	-5	46.3	6	39	-25	45.8
7	12	-5	46.3	7	23	-50	45.9
8	15	-5	46.3	8	50	-20	45.9
9	23	-20	46.5	9	6	-20	45.9
10	9	-30	46.5	10	34	-20	45.9

の中から，エラー率の低い局所最適点を 10 個リストアップしたものである．この表から，上位 10 個の局所最適点での単語エラー率の差は小さいことがわかる．この結果から，何らかの山登り法的な最適化手法を使っても，得られる結果は大域最適解とそれほど違わないことが期待できる．このような最適化手法については，5. で述べる．

3 N-best 候補の予備選択

3.1 音響スコアと言語スコアから総合スコアを計算すること

N-best 候補からの最適候補は，式 (5) に従って決定される．この過程についてもう少し考えてみよう．簡単のため，最初は挿入ペナルティについては考えないことにする．すると，ある一発話についての最適化過程は次のようになる．

$$\tilde{n}_i(W_L) = \underset{j}{\operatorname{argmax}}\, \boldsymbol{x}_A(i,j) + W_L \boldsymbol{x}_L(i,j) \qquad (7)$$

この計算において，総合スコア

$$\boldsymbol{x}(i,j) = \boldsymbol{x}_A(i,j) + W_L \boldsymbol{x}_L(i,j) \qquad (8)$$

が計算され，そのスコアが最も高い候補が最終的な結果として選ばれる．式 (8) は線型なので，$\boldsymbol{x}$ の計算は，$(\boldsymbol{x}_A, \boldsymbol{x}_L)$- 平面から平面 $\boldsymbol{x} = \boldsymbol{x}_A + W_L \boldsymbol{x}_L$ への射影とみなすことができる．これを図 3に示す．総合スコアが最も高い候補を選ぶことは，図中の射影された平面上で最も高い位置にある点を選ぶことと等価である．

ここで，平面 $\boldsymbol{x} = \boldsymbol{x}_A + W_L \boldsymbol{x}_L$ 上にあり，射影された全ての点を含む最小の多角形を考えよう．射影された点は傾いた平面上にあるので，その中で最も高い位置にある点は，明らかに多角形の頂点の一つである．したがって，最終結果として選ばれる候補は，$(\boldsymbol{x}_A, \boldsymbol{x}_L)$- 平面上の多角形の頂点でなければならない．

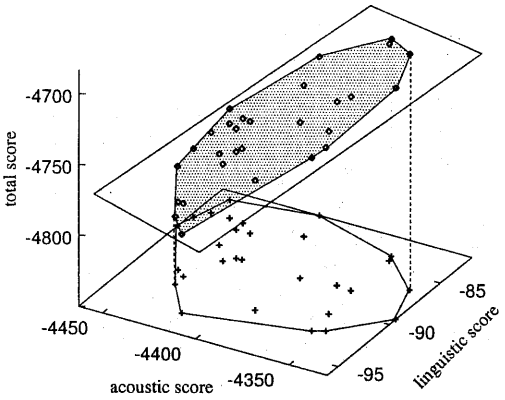

図 3: 射影された点を全て含む最小の多角形

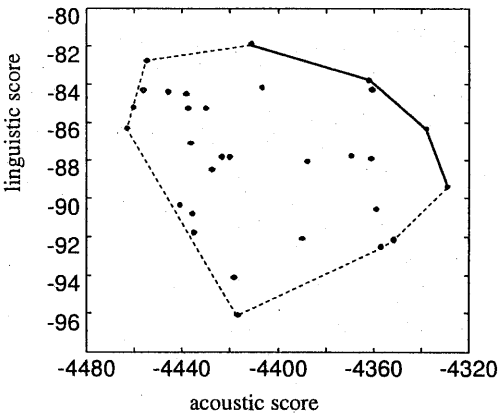

図 4: 多角形上で有効な候補

このことから，n-best 候補の数は，多角形の頂点の数にまで減らすことができる．なぜなら，多角形の頂点でない点は最終的に最大のスコアを持ち得ず，それゆえ最適化結果に全く影響を与えないからである．

さらに，W_L が正であることを考えると，さらに候補数を減らすことができる．W_L が正であるとすると，図 4において，点線上の候補は選ばれることがない．したがって，n-best 候補の数は，多角形の最も上の点と最も右の点を結ぶパス上の点の数に制限される．

3.2 挿入ペナルティの扱い

これまでの議論では，簡単のため挿入ペナルティを無視していた．挿入ペナルティを考慮すれば，$(\boldsymbol{x}_A, \boldsymbol{x}_L, n)$- 空間上での多面体を考慮する必要がある．このとき，n-best 候補数は，多面体の頂点の数まで減らすことができる．しかし，与えられた点を全て含む多面体を決定するのは容易ではないので，ここ

ではもっと簡単な方法を考える．単語数 n は離散値であり，一つの発話に対する候補の中ではそれほど変動しないことから，各単語数ごとに，前述の予備選択をすることができる．以下では，各単語数ごとの予備選択結果を合わせることで，挿入ペナルティに対する最適性を保証する．

3.2.1 多角形の決定法

次に，この多角形を決定するアルゴリズムについて考える．ここでは，次のようなアルゴリズムを用いた．このアルゴリズムは，$O(PV)$ の計算量を持つ．ここで，P は候補点の数，V は多角形の頂点数である．なお，この予備選択は，パラメータの最適化の前に一度だけ行えばよい．

P: 点の集合
$\theta_{a,b}$: 2 つのベクトル a と b の角度

set $Q \leftarrow \emptyset$
点 $p \leftarrow \mathrm{argmax}_{p \in P}\{y | p = (x, y)\}$
点 $p_e \leftarrow \mathrm{argmax}_{p \in P}\{x | p = (x, y)\}$
ベクトル $v \leftarrow (1, 0)$
loop
  $q \leftarrow \mathrm{argmax}_{q \in P} \cos\theta_{v, q-p}$
  もし $q = p_e$ ならば Q が解となる
  $v \leftarrow q - p$
  P から q を削除
end loop

3.2.2 実験

N-best 候補からの予備選択実験を行った．N-best 候補は 2.3で用いたものと同じである．その結果を次に示す．

	男性	女性	計
発話数	389	295	684
元の候補数	35030	26941	61971
選択された候補数	4333	3331	7664

N-best 候補の 87% が削減され，しかも最適化結果は全く同じであった．この予備選択を使うことで，最適化に必要な時間は 1/8 に短縮された．

4 頑健なパラメータの設定法

通常，開発テストセットからパラメータを決定する際に，そのパラメータがどの程度頑健であるかはあまり意識されないことが多い．しかし，2 節の結果から，より頑健なパラメータを選ぶ余地があることが示された．それでは，どのようにして頑健なパラメータを選んだらよいであろうか．ここでは，次のアルゴリズムを試してみた．

開発テストセット S を $S_1, S_2, \ldots, S_m$ に分割する．
$i = 1, \ldots, m$ について，
  すべての (W_L, P_I) の組みあわせについて，単語エラー率 $E(W_L, P_I; S_i)$ を計算する．

$i = 1, \ldots, m$ についての $E(w, p; S_i)$ の平均と分散を $\mu(w, p)$ と $\sigma(w, p)$ とする．
$\hat{w}, \hat{p} = \mathrm{argmin}_{w,p}\, \mu(w, p) + k\sigma(w, p)$ として $\hat{w}, \hat{p}$ を決定する．

ここで k は適当な定数である．

通常のパラメータ選択と，ここで述べたアルゴリズムを使う方法との比較実験を行った．最初に，これまで用いたのと同一の n-best のデータを，発話番号の末尾の数字によって $\{01, 23, 45, 67, 89\}$ の 5 つのサブセットに分ける．それぞれのサブセットの発話数と単語数は以下の通りである．

	01	23	45	67	89
発話数	140	150	139	131	124
単語数	1303	1408	1251	1283	1131

次に，これらのサブセットのうち 4 つを用いてパラメータを決定し，それを使って残りのサブセットを評価した．評価に使うセットを変えながら，5 種類の実験を行った．このときの結果を次の表に示す．

評	$k=0$			$k=1$			$k=2$		
価	W_L	P_I	WER	W_L	P_I	WER	W_L	P_I	WER
89	23	-20	47.57	8	-20	46.42	8	-20	46.42
67	15	-15	48.95	15	-15	48.95	8	-20	47.31
45	15	-35	48.52	7	-25	48.20	11	-35	49.56
23	9	-20	43.96	8	-20	44.18	49	35	44.11
01	9	-20	45.51	8	-20	45.66	8	-25	45.51
平均			46.82			46.63			46.52

$k = 2$ の場合の結果では，従来法による最適化結果 ($k = 0$) と比べて，0.31 ポイント低い単語エラー率が得られた．この手法の有効性を確かめるため，さらに異なる分割について実験を行なった．

- 性別と発話の偶数・奇数による 4 つのサブセット (4.1)
- 発話番号による 4 つのサブセット (4.2)
- 発話番号による 5 つのサブセット (5; 前述のものと同じ)
- 発話番号による 6 つのサブセット (6)

それぞれの分割に対する単語エラー率を次に示す．

data	original	k				
	WER	0	1	2	3	4
4.1	46.85	46.85	46.66	46.72	46.49	46.86
4.2	46.80	46.88	46.88	46.85	46.73	46.89
5	46.82	46.82	46.63	46.51	46.60	46.60
6	46.68	46.74	46.89	46.69	46.36	46.36

$k = 3$ の場合，すべての条件で性能の向上が見られたが，その差はあまり大きくなかった．

表 3: 提案法による最適化結果

偶数データ

探索手法	探索点数	W_L	P_I	WER
全探索	1050	9	-10	45.9
近隣サーチ	25	9	-20	46.2

奇数データ

探索手法	探索点数	W_L	P_I	WER
全探索	1050	16	-50	45.6
近隣サーチ	17	6	-20	45.9

同じ実験を，trigram言語モデルについても行った．このモデルは，Switchboardコーパスから作成したものである．単語エラー率は次のようになった．

data	original	k				
	WER	0	1	2	3	4
4.2	45.34	45.34	45.42	45.35	45.45	45.40
5	45.15	45.15	45.51	45.54	45.51	45.51
6	45.39	45.42	45.92	45.84	46.08	46.33

Trigramの場合，このアルゴリズムで性能を改善することはできなかった．提案法がどのような場合に有効か，さらに検討を要する．

5 パラメータ設定の効率的な探索法

5.1 近隣サーチによる方法

2節で述べた通り，もし1ポイント程度の性能差が許容できる場合は，もっと効率的なパラメータ設定法が使える可能性がある．この節では，近隣サーチによる最適化手法について述べる．基本的な考え方としては，すべてのパラメータの組みあわせを試すかわりに，現在の点の8近傍だけを試す．もし，現在の点がその8近傍における単語エラー率よりも低ければ，現在の点はその付近での局所最適解であることがわかる．そうでなければ，8近傍の中で単語エラーが最低の点を新たな現在の点とする．

5.2 実験

提案法の効率と性能を調べる実験を行った．これまで用いたn-bestデータを，偶数番目と奇数番目のデータに分け，それぞれについて最適化を行った．このとき，$W_L = (1, 2, \ldots, 50)$ と $P_I = (-50, -45, -40, \ldots, 45, 50)$ のすべての組みあわせについて最適化を行った．次に，近隣サーチに基づく方法を試した．このとき，初期値を $(5, -10)$ とし，$(\Delta w, \Delta p) = (1, 5)$ とした．結果を表3に示す．

これらの結果と表2から，近隣サーチによる結果では，偶数データに対して5番目に良い結果を返し，奇数データに対して9番目に良い結果を返していることがわかる．これらの結果は最適な単語エラー率から0.3ポイント高いだけであるのに対し，探索された点は全探索の場合の1～2%だけであった．

6 結論

本稿では，言語重みと挿入ペナルティに関するいくつかの問題に関して考察し，新しいアルゴリズムを提案した．最初に，パラメータ最適結果について調べ，次の三つの事実を発見した．

1. 最適な言語重みと挿入ペナルティは，開発テストセットに依存する．したがって，どんな入力にも最適であるようなパラメータ設定を見つけることは不可能である．
2. パラメータ設定が入力に依存するので，言語重みと挿入ペナルティに関して頑健性の問題が存在する．これによる単語エラー率の上昇は，1～2ポイントと見積もられる．
3. パラメータ空間には多くの局所最適点が存在するため，効率的に大域最適点を探すことが難しい．しかし，それぞれの局所最適点における単語エラー率は，大域最適点におけるそれとあまり違わない．

次に，n-best候補の予備選択を行うアルゴリズムを提案した．このアルゴリズムは，最適化結果を全く変えずに，最適化時間の87%を削減する．さらに，より頑健なパラメータ設定を探すためのアルゴリズムを提案した．実験結果から，bigram言語モデルに関しては，提案法は従来法よりもわずかに良い結果を与えたが，trigramに対しては改善が得られなかった．最後に，近隣サーチに基づく最適化法を提案した．この手法は最適化の時間を劇的に短縮し，準最適な結果を与える．

言語モデルと挿入ペナルティの最適化に関しては，まだ多くの問題が残されている．言語重みと挿入ペナルティがなぜ必要で，それを決める要因が何なのかはまだ十分理解されていない．おそらく言語スコアと音響スコアの平均と分散の差が関係していると思われるが，直接の証拠は得られていない．これらの関係について，実際のデータと整合性のある理論が発見されれば，もっと効率的なパラメータ設定法が開発できるであろう．

謝辞

研究の機会を与えていただいた，ワシントン大学のProf. Mari Ostendorf, およびデータを提供していただいたDr. Michiel Bacchianiに感謝します．

参考文献

[1] A. Ogawa *et al.* "Balancing Acoustic and Linguistic Probabilities", Proc. ICASSP98, pp. 181–184 (1998)