ACOUSTICAL LETTER

Fast optimization of language model weight and insertion penalty from n-best candidates

Akinori Ito[1,*], Masaki Kohda[2,†] and Shozo Makino[1,‡]

[1]*Graduate School of Engineering, Tohoku University, Aramaki aza Aoba 6–6–5, Sendai, 980–8579 Japan*
[2]*Faculty of Engineering, Yamagata University, Jonan 4–3–16, Yonezawa, 992–8510 Japan*

(*Received 4 February 2005, Accepted for publication 23 February 2005*)

Keywords: Language model weight, Insertion penalty, Optimization, N-best hypothesis
PACS number: 43.72.Ne [DOI: 10.1250/ast.26.384]

1. Introduction

Almost all LVCSR (Large Vocabulary Continuous Speech Recognition) systems have several parameters, such as the language model weight and insertion penalty. In many cases, these parameters are determined empirically or through preliminary recognition tests. There have been a few efforts to treat these parameters theoretically [1,2], but the optimum parameter still cannot be determined automatically. To optimize these parameters through experiments, we must carry out recognition tests for all points in the parameter space, which is quite time-consuming. N-best-rescoring-based optimization is often employed to expedite the optimization process, but it still takes considerable time when the number of n-best candidates is large.

We describe a new idea for preselecting n-best candidates to make n-best-based parameter optimization faster. Although it is a very simple idea, the method enables the number of n-best candidates to be reduced by more than 90% and makes the optimization process about 9–28 times faster.

2. N-best-candidate-based parameter optimization

When recognized candidates are obtained from a speech recognizer, they are evaluated using their scores. Usually a candidate has an acoustic score x_A and a linguistic score x_L. They are often logarithms of probabilities $P(x|w)$ and $P(w)$, respectively where x is the acoustic signal and w is a corresponding word sequence.

To compare candidates using a one-dimensional measure, a total score is calculated from acoustic and linguistic scores. The number of words in a candidate sentence is also often taken into account. The most popular way of combining the scores is linear combination:

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

where n_w is the number of words in a candidate. For comparison, only the relative magnitude of the score is significant. Therefore, *assuming a_1 to be positive*, the following score can be used:

$$x = x_A + \frac{a_2}{a_1} x_L + \frac{a_3}{a_1} n_w \tag{2}$$

$$= x_A + w_L x_L + w_I n_w, \tag{3}$$

where w_L and w_I are called the *language model weight* and the *word insertion penalty*, respectively. The optimization of the parameters is the process of determining w_L and w_I that give the lowest word error rate.

The language model weight and insertion penalty can be optimized by carrying out recognition experiments for all combinations of (w_L, w_I). However, doing "real" recognition for all parameter combinations takes a very long time. To make the optimization faster, word-graph-based optimization [3] or n-best-candidate-based optimization [4] is often used. First, calculate the word graph or n-best recognition result for each utterance using arbitrary parameter values. Strictly speaking, the quality of the candidates depends on the initial parameter values used in the first decoding process. To make the process more accurate, the generation of candidates can be iterated [3]. However, the effect of the initial parameters can be small if the number of candidates is sufficiently large. After the word graphs or n-best candidates are obtained for all utterances, parameters are optimized by rescoring the candidates.

The n-best-based method can be formulated as follows. Let us define the following symbols.

M	number of utterances
$x(i, j, w_L, w_I)$	total score of j-th candidate of i-th utterance for language model weight w_L and insertion penalty w_I
$x_A(i, j)$	acoustic score of j-th candidate of i-th utterance
$x_L(i, j)$	linguistic score of j-th candidate of i-th utterance
$n_w(i, j)$	number of words in j-th candidate of i-th utterance
$E(i, j)$	number of substituted/deleted/inserted words of j-th candidate of i-th utterance

The typical optimization of parameters w_L and w_I is as follows:

*e-mail: aito@fw.ipsj.or.jp
†e-mail: kohda@yz.yamagata-u.ac.jp
‡e-mail: makino@ecei.tohoku.ac.jp

$$x(i, j, w_{\mathrm{L}}, w_{\mathrm{I}}) = x_{\mathrm{A}}(i, j) + w_{\mathrm{L}} x_{\mathrm{L}}(i, j) + w_{\mathrm{I}} n_w(i, j) \quad (4)$$

$$\tilde{o}_i(w_{\mathrm{L}}, w_{\mathrm{I}}) = \operatorname*{argmax}_j x(i, j, w_{\mathrm{L}}, w_{\mathrm{I}}) \quad (5)$$

$$\tilde{w_{\mathrm{L}}}, \tilde{w_{\mathrm{I}}} = \operatorname*{argmin}_{w_{\mathrm{L}}, w_{\mathrm{I}}} \sum_{i=1}^{M} E(i, \tilde{o}_i(w_{\mathrm{L}}, w_{\mathrm{I}})). \quad (6)$$

$\tilde{o}_i(w_{\mathrm{L}}, w_{\mathrm{I}})$ is the optimum candidate of the i-th utterance under parameter $(w_{\mathrm{L}}, w_{\mathrm{I}})$.

3. Preselection of n-best candidate

The optimum candidate among n-best candidates is determined according to Eq. (5). Let us further consider the determination process. For simplicity, the insertion penalty is ignored in the discussion here. Therefore the optimization process becomes

$$\tilde{o}_i(w_{\mathrm{L}}) = \operatorname*{argmax}_j \{x_{\mathrm{A}}(i, j) + w_{\mathrm{L}} x_{\mathrm{L}}(i, j)\}. \quad (7)$$

In this process, the total score,

$$x(i, j, w_{\mathrm{L}}) = x_{\mathrm{A}}(i, j) + w_{\mathrm{L}} x_{\mathrm{L}}(i, j), \quad (8)$$

is calculated and the candidate with the highest total score is chosen as the final result. As Eq. (8) is linear, the calculation of x can be regarded as the projection of points from the $(x_{\mathrm{A}}, x_{\mathrm{L}})$-plane to the plane $x = x_{\mathrm{A}} + w_{\mathrm{L}} x_{\mathrm{L}}$. Figure 1 shows an example of the projection. Choosing the candidate with highest total score is equivalent to choosing the highest point among the projected points.

Let us consider a polygon on the plane $x = x_{\mathrm{A}} + w_{\mathrm{L}} x_{\mathrm{L}}$, that is, the smallest polygon which contains all the projected points. As the projected points are on a tilted plane, it is obvious that the highest point is on the edge of the polygon (Fig. 2). Therefore, the candidates that can be chosen as the final result must be on the edge of the polygon on the $(x_{\mathrm{A}}, x_{\mathrm{L}})$-plane.

The above fact indicates that the number of n-best candidates can be reduced to the number of points on the edge of the polygon, because the candidates inside the polygon (and not on the edge of the polygon) are never chosen as the recognition result and they do not affect the optimization process.

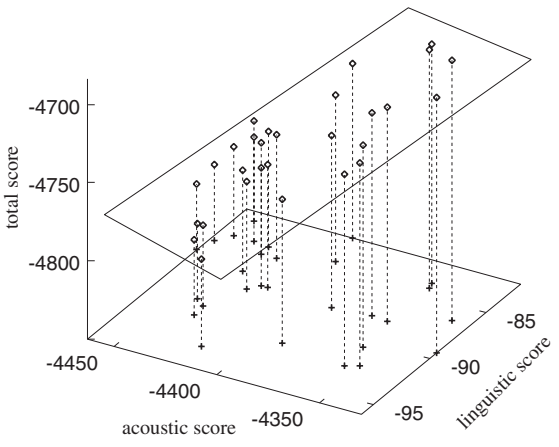

Fig. 1 Calculating total score from acoustic and linguistic score.

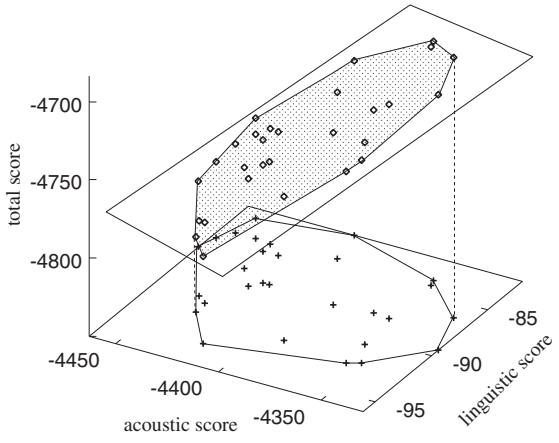

Fig. 2 The smallest polygon that contains all the projected points.

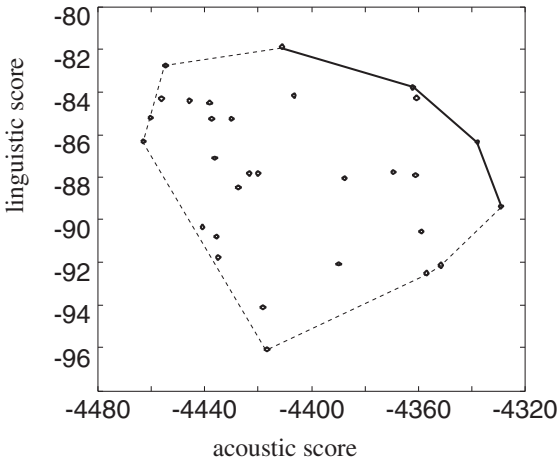

Fig. 3 Valid candidates on the edge of the polygon.

Considering that w_{L} is positive, further reduction of the number of candidates can be achieved. In Fig. 3, candidates on the broken lines are never chosen if w_{L} is positive. Therefore, n-best candidates can be restricted to the points on the path of the polygon between the uppermost point and the rightmost point.

Considering the insertion penalty, we must determine a polyhedron in $(x_{\mathrm{A}}, x_{\mathrm{L}}, n_w)$-space. N-best candidates are reduced to the number of points on the surface of the polyhedron. However, it is not easy to determine the smallest polyhedron that contains all points. As n_w is discrete and does not vary greatly among the candidates of one utterance, it will be sufficient to preselect for each n_w-word candidate.

In the later experiments, the algorithm shown in Fig. 4 was used for preselection. The algorithm has a computational complexity of $O(PV)$, where P is the number of points and V is the number of vertices of the polygon. Note that preselection is executed only once before parameter optimization.

4. Experiment

An experiment was carried out to confirm the effect of preselection. Experimental conditions are shown in Table 1. In this experiment, N-best ($N = 5$–1,000) candidates were

P: set of points
$\theta_{a,b}$: angle between two vectors a and b

point $p \leftarrow \mathrm{argmax}_{p \in P}\{y | p = (x, y)\}$
point $p_e \leftarrow \mathrm{argmax}_{p \in P}\{x | p = (x, y)\}$
set $Q \leftarrow \{p\}$
remove p from P
vector $v \leftarrow (1, 0)$
loop
  $q \leftarrow \mathrm{argmax}_{q \in P} \cos\theta_{v, q-p}$
  add q to Q
  if $q = p_e$ then Q becomes the answer. stop.
  $v \leftarrow q - p$
  remove q from P
  $p \leftarrow q$
end loop

Fig. 4 An algorithm to determine the polygon.

Table 1 Experimental conditions.

Acoustic model [5]	PTM 64 mixture, 3,000 states, gender-independent
Language model [5]	backoff bigram/trigram, 20 k vocabulary, learned from 75 months' worth of issues of Mainichi Shimbun
Test data	100 sentences by 10 male speakers (from ASJ-JNAS)
w_{L} search space	$0.0, 0.5, 1.0, 1.5, \ldots, 30.0$
w_{I} search space	$-30, -29, \ldots, 29, 30$

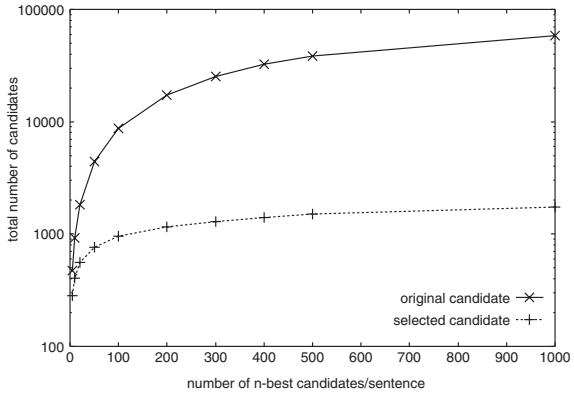

Fig. 5 Number of total candidates and preselected candidates.

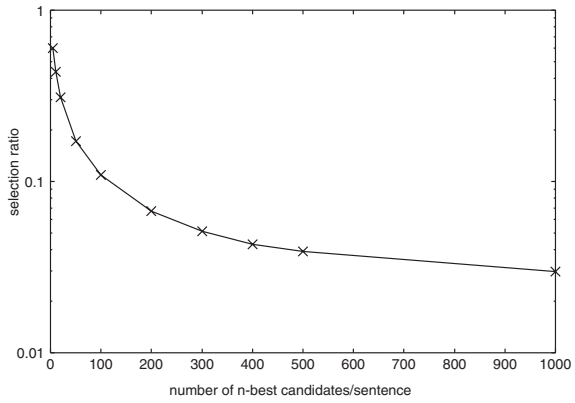

Fig. 6 Preselection ratio.

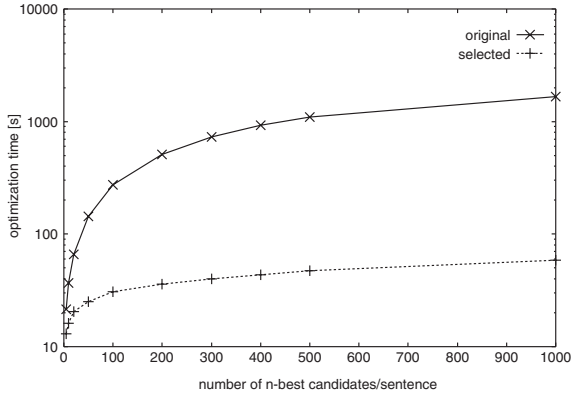

Fig. 7 Optimization time with/without preselection.

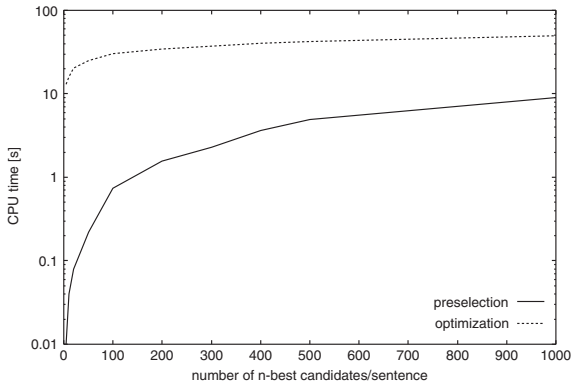

Fig. 8 Preselection and optimization times.

calculated from 100 utterances. The preselection method was applied to the candidates. The number of total candidates in the 100 utterances and the number of selected candidates are shown in Fig. 5. When up to 1,000-best candidates were used, 58,159 candidates were generated and only 1,733 candidates remained after preselection. Figure 6 shows the selection ratio (number of selected candidates divided by total number of candidates). From this result, it was found that the preselection method was more effective when the number of candidates per utterance was large. When 1,000-best candidates were used, the number of candidates was reduced by 97%. Figure 7 shows the CPU times taken for optimization with or without preselection. The experiment was performed on a PC with 1.8 GHz PentiumIII™ processor. The processing time with preselection includes the time taken for preselection. The processing time was almost proportional to the number of candidate. The optimization time with preselection was about 9 times faster than that without preselection under the 100-best condition, and 28 times faster under the 1,000-best condition. Figure 8 shows the CPU time taken for

preselection and the following optimization procedure. The result indicates that the preselection procedure takes only 2.5% (at 100-best) to 18% (at 1,000-best) of the CPU time compared with the optimization procedure.

5. Conclusion

An algorithm for preselection of n-best candidates is proposed. The use of this algorithm makes the optimization time 9–28 times faster without changing the optimization result.

References

[1] A. Ogawa, K. Takeda and F. Itakura, "Balancing acoustic and linguistic probabilities," *Proc. ICASSP 98*, pp. 181–184 (1998).

[2] Y. Horibe, N. Minematsu and S. Nakagawa, "Theoretical/experimental investigation of balance between acoustic model likelihood and language model likelihood," *IPSJ SIG Notes*, Vol. 2000, No. 54, pp. 67–72 (2000).

[3] M. Katoh, T. Saiin, A. Ito and M. Kohda, "Optimization of the parameter set for word graph generation," *IPSJ SIG Notes*, Vol. 2000, No. 119, pp. 107–112 (2000).

[4] A. Stolcke, Y. König and M. Weintraub, "Explicit word error minimization in n-best list rescoring," *Proc. Eurospeech '97*, pp. 163–166 (1997).

[5] T. Kawahara, A. Lee, T. Kobayashi, K. Takeda, N. Minematsu, S. Sagayama, K. Itoh, A. Ito, M. Yamamoto, A. Yamada, T. Utsuro and K. Shikano: "Free software toolkit for Japanese large vocabulary continuous speech recognition," *Proc. ICSLP 2000*, pp. 476–479 (2000).