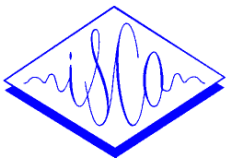

3rd European Conference on
Speech Communication and Technology
EUROSPEECH'93
Berlin, Germany, September 19-23, 1993

Dictation System Using Inductively Auto-Generated Syntax

Shoichi Matsunaga[1], *Tomokazu Yamada*[2], and *Kiyohiro Shikano*[2]

[1]ATR Interpreting Telecommunications Research Laboratories
Seika-cho, Soraku-gun, Kyoto Japan

[2]NTT Human Interface Laboratories
Midori-cho, Musashino-shi, Tokyo Japan

ABSTRACT

This paper describes a Japanese dictation system that can effectively deal with an unlimited vocabulary. An approach that automatically generates a character n-gram syntax, from both the original training text and newly generated text is proposed. Each sentence phrase of the newly generated text that is not included in the training text is created by using the training text and the character trigram model of that text. About one-third of the search space not covered by the training text is covered by the newly generated text, showing the effectiveness of the text auto-generation approach. Furthermore, compared with the common beam-search technique, the proposed search technique requires about three-fourths less processing time and allows more accurate recognition.

Keywords: *Speech Recognition, Language Modeling, Searching.*

10.21437/Eurospeech.1993-478

1 Introduction

One goal of automatic speech recognition is a device capable of transcribing speech into text. For this goal, a large text database is usually used to generate a stochastic language model, such as a word trigram model, thus allowing accurate recognition[1]. For example, IBM used approximately 10 million words for training[2]. In contrast, the search space for recognition is defined from a given syntax, and the syntax is usually very large relative to the training text. Thus, Bell Laboratories generated good training sets for speech material heuristically[3].

The large search space is the main problem in recognizing continuous speech, as it greatly increases processing time. We found the same problem in developing our dictation system[4], which deals with an unlimited vocabulary. Deterministic constraints on the search space would make recognition faster, if a sufficiently complete text database were available, but it would be impossible to collect such a database covering every field.

Our approach is therefore to generate new text inductively by using a training text and its stochastic model. This new text is not included in the training text and is generated to have relatively high probability of occurrence based on the stochastic model of the training text. We also propose a fast search technique that constrain search space by using the training text and newly generated text for continuous speech recognition.

2 Auto-Generation of Text

Figure 1 shows each of the abstract search spaces for speech recognition. The space S_h is used in daily conversation or writing, the space S_l is defined from the training text, and S_a is every possible sequence of all words. The space S_a, which was used in our dictation system, is too large, and S_h is impossible to precisely define. Our approach searches S_s, which is the union of S_e (defined from the newly generated text) and S_l ($S_s = S_e \cup S_l$). Although S_e includes some illegal sentences, it should contain the most likely sentences in $S_h - S_l$.

Assume a training text consisting of M sentences. The m-th ($1 \leq m \leq M$) sentence consists of N words, and the dividing point of the sentence is between the n-th and $(n+1)$-th words. The likelihood of dividing the sentence is defined as $\gamma(n,m) = \alpha(n,m) + \beta(n+1,m)$, where α is the occurrence likelihood of the sequence of words 1 to n, and β is the likelihood of words $(n+1)$ to N. For

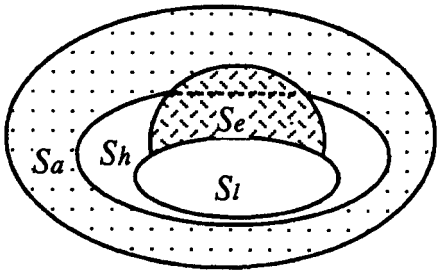

Figure 1. Abstract search spaces.
Search space: for example, duplets or triplets of words.

example, using the trigram model,

$$\alpha(n,m) = \frac{1}{n}\sum_{i=1}^{n} \log P(w_i|w_{i-1}, w_{i-2}) \qquad \text{and}$$

$$\beta(n+1,m) = \frac{1}{N-n}\sum_{j=n+1}^{N} \log P(w_j|w_{j+1}, w_{j+2}),$$

where w_i is the i-th word in the sentence, and $P(w_i|w_{i-1}, w_{i-2})$ is the conditional occurrence probability of w_i after w_{i-2}, w_{i-1}. Thus α is the averaged forward likelihood of a word sequence, and β is the backward likelihood. The more appropriate a dividing point is, the higher its γ value. The set of dividing points is denoted V. The inductive auto-generation algorithm is as follows:

1. Find (n_{max}, m_{max}) giving the maximum value of $\max_{n,m \in V} \gamma(n,m)$. In Figure 2(a), for example, it is hypothesized that n_{max} is 2 and m_{max} is 1. When the value $\gamma(n_{max}, m_{max})$ is less than a threshold γ_{th1}, the generation process is terminated.

2. Detect all sentences, whose likelihoods $\gamma(n_{max}, m)$ are greater than a threshold γ_{th2}, and that have the same first half, $w_1 \cdots\cdots w_{n_{max}}$. Each second half of the detected sentences, $w_{n_{max}+1} \cdots\cdots w_{N(m)}$, is stored in Table B.

3. Detect all sentences $(\gamma(n_{max}, m) > \gamma_{th2})$ that have the same second half, $w_{n_{max}+1} \cdots\cdots w_{N(m_{max})}$, and store the first halves of those sentences in Table A. Figure 2(b) shows "a pencil" in Table B and "That is" in Table A.

4. Generate new sentences by combining elements in Table A and Table B. In Figure 2 (c) "That is a pencil" is newly generated.

5. Set $V = V - \{(n_{max}, m_{max})\}$, and go back to step 1.

A Japanese sentence consists of several phrases, and each phrase consist of several Japanese ("Kana") and Chinese ("Kanji") characters. Each Japanese character has semantic meaning and is composed of about two syllables. When the above algorithm is applied to Japanese phrase text, m shows the number of phrases, n shows the number of characters and w_i is i-th character in a phrase.

3 Experiments on Generating New Text

A preliminary test on text generation was carried out using 1000 Japanese phrases concerning medical diagnoses. The dividing points were character boundaries. A native Japanese determined that 74% of the phrases in the generated text were syntactically and semantically correct. The generated text was also verified by comparing it with a text consisting of 72000 phrases: 37% of the phrases in the generated text were included in the larger text. These results show that our text generation algorithm is promising.

We then tested the coverage of the spaces S_l (training text) and S_s (generated text) – that is, what fraction of all duplets and triplets in the test text are in these search spaces (duplets and triplets)? Three training texts – consisting of 5000, 10000, and 15000 phrases – were prepared, and the test text was 12000 phrases. As listed in Table 1, about one-third of the search space not covered by the training text is covered by the newly generated text, keeping perplexities almost equal. This shows the effectiveness of inductive text generation.

We then compared the number of triplets in the generated phrases to that in the newly collected phrases in the 12000-phrase test text (Figure 3). The horizontal axis x indicates the number of phrases generated from 15000 phrases or the number of newly collected phrases in addition to the original 15000 phrases. The vertical axis shows { (the number of character triplets in the 15000+x phrases) – (the number of triplets in the 15000 phrases) } or { (the number of triplets in the 15000 phrases and the newly generated x phrases) – (the number of triplets in the 15000 phrases) }. Though the triplets covered in text generation is less than that covered in new text collection, there is still good coverage in the text generation

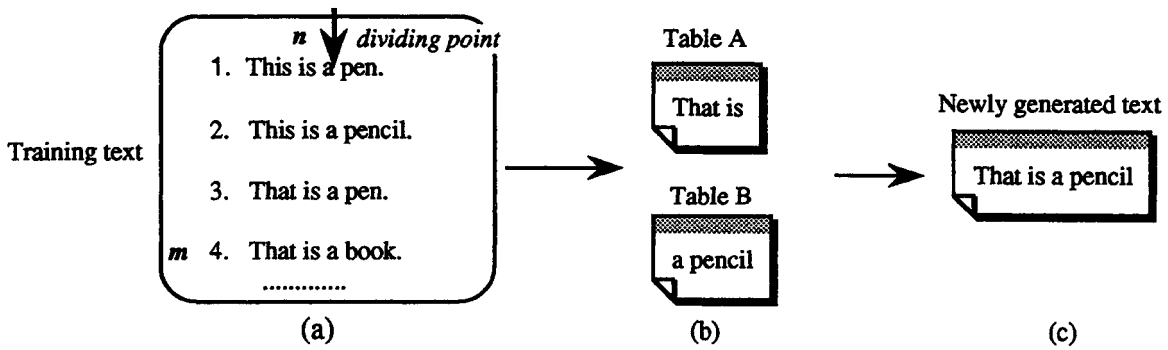

Figure 2. Example of text generation.

Table 1. The coverage (%) using character duplets and triplets, and character perplexities of bigram- and trigram-based covered sequences. (*perplexity*)

	search space	number of phrases in training text		
		5000	10000	15000
duplet (bigram)	S_l	94.8% (4.4)	97.0% (4.6)	97.6% (4.7)
	$S_s(=S_e \cup S_l)$	96.3% (4.5)	97.8% (4.7)	98.3% (4.7)
triplet (trigram)	S_l	90.0% (3.3)	93.6% (3.3)	94.6% (3.4)
	$S_s(=S_e \cup S_l)$	92.9% (3.3)	95.6% (3.3)	96.5% (3.4)

S_l: *defined from training text*

S_e: *defined from newly generated text*

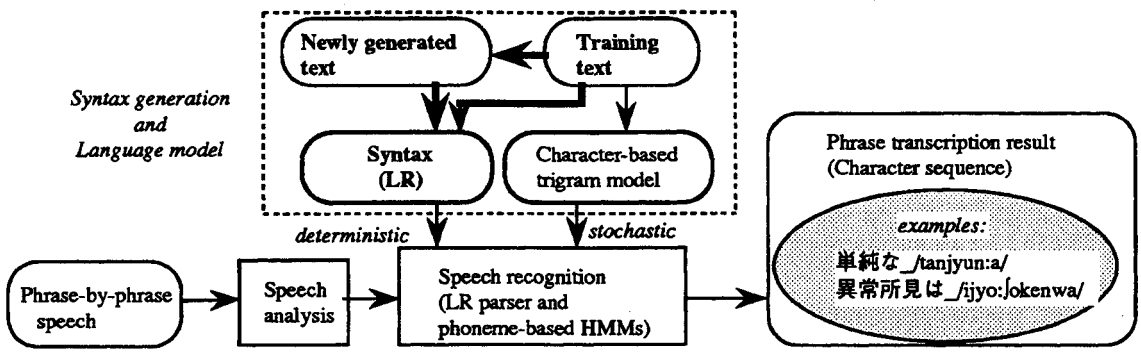

Figure 4. Dictation system using inductively auto-generated syntax

in spite of self-generation.

4 Continuous Speech Recognition Experiments

We have been developing a dictation system, dealing with unlimited vocabulary and various sentence structures[4]. The recognition process (Figure 4) of our dictation system is as follows: First, power, LPC cepstra and delta-cepstra are computed for each frame of the input speech and are fuzzy-vector-quantized. Next, based on phoneme-based hidden Markov models (HMMs) and a predictive LR parser[5], the phoneme sequence likelihood is calculated. A joint likelihood, combining acoustic and linguistic likelihoods derived from the HMMs and a character trigram model, is maximized to obtain the optimal recognition. Note that a beam search is used for rapid recognition. Beam width N roughly corresponds to the top–N candidates. In our first experiments, the beam width was 600. The character trigram model and the new text are generated from the training text of 15000 phrases, and the model is normalized using deleted interpolation[1].

In speaker-dependent recognition (Test 1), in which the search space was S_a (every possible sequence of all characters), the number of characters was 920, and the character perplexity for the test data was 4.54, 79.9% of 378 phrases (part of the test text) were recognized correctly.

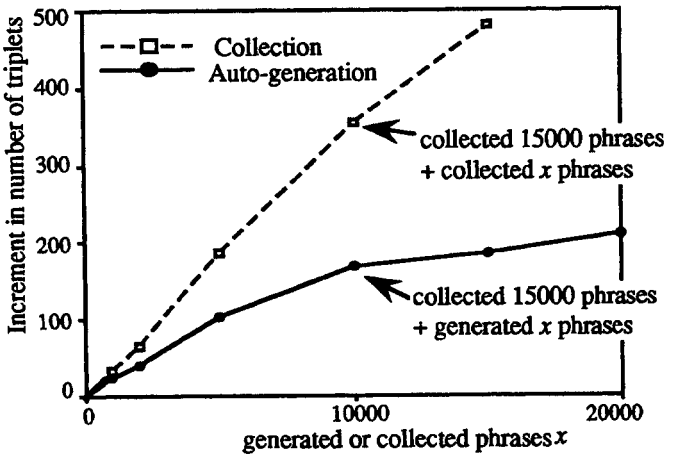

Figure 3. Increment in number of triplets for auto-generation and for collected phrases

We also tested recognition using the syntax generated by our approach, where the syntax (or search space) was defined from character quartets in the training text and the newly generated text. Recognition was nearly four times as fast as Test 1. Furthermore, the phrase recognition rate increased to 85.2%, showing the effectiveness of the inductive syntax-generation approach to dictation. These results are shown in Table 2.

Each normalized recognition time (%) using 222 phrases, where th beam width was 300, was shown in Figure 5 for the search space, S_a, character solo, duplet, triplet or quartet. This shows that reduction in processing time was remarkable where the search space was character triplet

Table 2. Phrase recognition rates and processing times.

	search space	normalized time	recognition rate	perplexity
Test 1 (previous method[4])	S_a	100%	79.9%	4.54*1
Test 2 (proposed method)	$S_s(=S_e \cup S_l)$	26%	85.2%	4.50*2

*1:*Deleted interpolation was used.* *2:*Flooring value of* 10^{-5} *was used.*

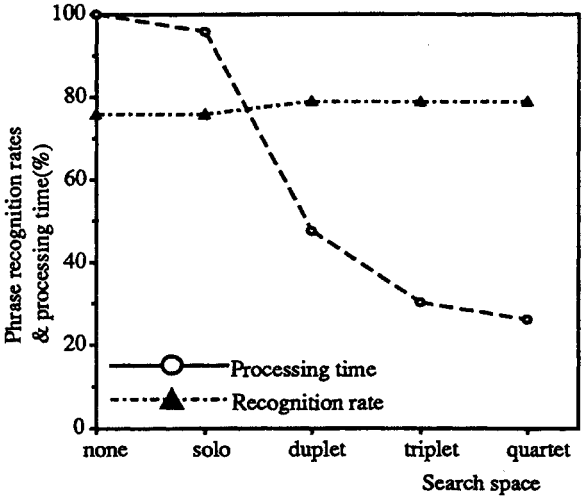

Figure 5. Relationship between search space and phrase recognition rate or processing time

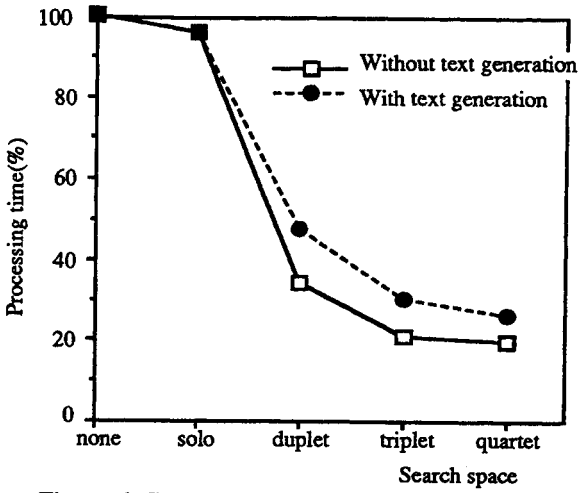

Figure 6. Comparison of processing time without text generation (15000 phrases) and with text generation (about 75000 phrases)

and character quartet. Finally, we compared the recognition time where the search space was defined from the 15000 phrases only, and defined from 15000 phrases plus newly generated text (almost 60000 phrases). As shown in Figure 6, the increase in processing time by using the newly generated text is not overly large. This shows that the amount of the generated text is not a large factor in recognition time.

5 Conclusion

This paper proposed inductive text generation and a constrained search space for continuous speech recognition. This approach not only reduces the processing time but also increases the recognition performance. According to the text-generation tests, about one-third of the search space not covered by the training text was covered by the newly generated text. Furthermore, according to the speech recognition experiments, compared with our previous beam-search used in the dictation system, the proposed search required about one-fourth the processing time and allows more accurate recognition.

To achieve higher coverage and faster recognition, we are currently implementing higher linguistic knowledge[6], such as part-of-speech and semantic tags, in the text generation process. The effect of inductive generation using these tags will be shown in the conference paper.

Acknowledgment

The authors would like to express their appreciation to Dr. Sadaoki Furui, NTT Human Interface Laboratories, for his invaluable guidance.

References

[1] L. R. Bahl, F. Jelinek, and R. L. Mercer,"A maximum likelihood approach to continuous speech recognition", *IEEE Trans.*, PAMI-5, 2, pp.179-180(1983)

[2] L. R. Bahl et al.,"A tree-based statistical language model for natural language speech recognition", *IEEE Trans.*, ASSP-7, 37, pp.1001-1008(1989)

[3] M. K. Brown et al.,"Training set design for connected speech recognition", *IEEE Trans.*, ASSP-6, 39, pp.1268-1281 (1991)

[4] T. Yamada, S. Matsunaga, and K. Shikano,"Japanese dictation system using character source modeling", *Proc. ICASSP-92*, pp.37-40 (1992)

[5] M. Tomita,"Efficient parsing for natural language: a fast algorithm for practical systems", Kluwer Academic Publishers (1986)

[6] R. Pieraccini et al., "A speech understanding system based on statistical representation of semantics", *Proc. ICASSP-92*, pp.193-196 (1992)