Japanese word segmentation using similarity measure for IR

Tomohiro Ozawa† Mikio Yamamoto† Kyoji Umemura†† Kenneth W. Church†††

† University of Tsukuba, {ozawa@milab., myama@}is.tsukuba.ac.jp

†† Toyohashi University of Technology, umemura@tutics.tut.ac.jp

††† AT&T Labs - Research, kwc@research.att.com

Abstract

It remains an open question what are the best units for IR, in particular for Asian languages; words, phrases, bigrams or ngrams. Our proposal is that the best units are what maximize a similarity measure between a query and a document. That is, in this framework, the IR system should have different representations of a query for each document. We develop the method which segments a query into unigrams, bigrams and arbitrary length ngrams using a similarity measure such as tf*idf as the criteria for the segmentation. Experimental results show that the method takes advantage of technical terms which tend to be longer than bigrams, and integrates the advantages of the word-based method and the ngram-based method without drawbacks of both.

Keywords: character-based IR method, ngram, word segmentation, suffix array

1. Introduction

There is an analogy between Information retrieval (IR) and the pattern recognition such as speech recognition. The task of acoustical model of the speech recognition system is to output a ranked list of phonemes based on an acoustic similarity between input speech and stored phoneme models[1]. An IR system outputs a ranked list of documents based on a similarity measure between an input query and stored documents such as the vector space model. Since speech has many variations of the pattern in its time structure, Hidden Markov Model (HMM) or Dynamic Time Warp (DTW) methods are used in order for the speech comparison to be meaningful. In the methods, the similarity score is defined as the best score in possibilities of alignment between the input and the stored patterns. We think that text has the same difficulty of matching; queries and documents have many variations like speech.

In this paper, we apply the idea of the best matching for each query and document to the transformation process of IR from a query to a representation. In IR for Asian languages, it remains an open question what are the best units; words, phrases, bigrams, ngrams or their combinations[2,3,4,5]. Our proposal is that the best units are different for each query and document. Like speech recognition, the similarity is defined as the best score in all possible representations of an input query. We will develop the method which segments an input query into arbitrary length ngrams as a representation of the query. The segmentation results maximize the similarity score for each document.

In the next section, the bigram-based method is compared to the word-based method. We will show that for a type of queries the bigram-based method is better, but for the other type of queries the word-based method is better. We intend to develop the method integrating the advantages of both methods without the drawbacks. Firstly, we will introduce the 'bigram segmentation' method which segments an input query to the one best sequence of unigrams and bigrams using the tf*idf measure as a segmentation criteria. Secondary, we apply the idea of the best and different segmentation for each document to the method; we will call this the 'adaptive bigram segmentation' method. Lastly, the bigram will be extended to arbitrary length ngrams. The last method takes advantage of technical terms which tend to be longer than bigrams, without missing documents which include the variations of the term.

2. Word and bigram methods

Recent studies have shown that the bigram-based method is comparable to or better than the word-based method for IR in Chinese and Korean[2,3,4]. We compared both methods using a normal tf*idf measure (see Table 1 in Section 4 in detail) and the test collection NTCIR-1 (preliminary version)[6] to be sure if the observation is also true for Japanese and this collection. We focused on the performance for each query. Figure 1 shows the 11pt. average precision of two methods for each query. Query ID's are sorted in order of precision of the word-based method. We can conclude that the advantage of each method is case by case. For some queries the bigram method is better than the word method, but vice versa for some other queries.

The bigram-based method is clearly worse in queries 25, 14, 23 and so on, which include longer technical terms than bigram. For example, query 25 includes a technical term 'LFG' which is an abbreviation of "Lexical

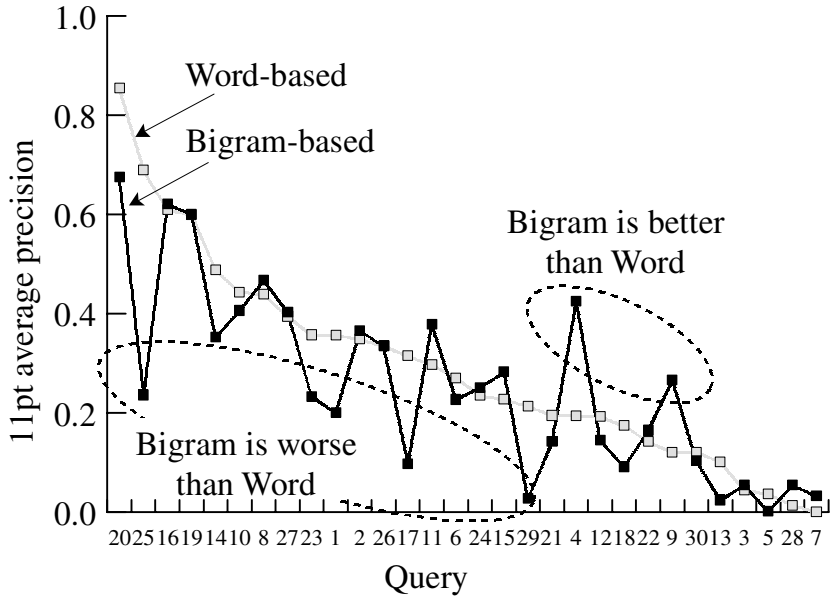

Figure 1: Word is better than bigram for some queries, but bigram is better than word for some other queries.

terms	total term freq.	doc. freq.
LFG	88	27
LF	1127	441
FG	1231	459
文書画像	549	270
書画	578	287
画像理解	227	147
像理	240	158
語彙機能文法	25	17
彙機	25	17
語彙	873	486
機能	50,676	29,569
文法	2,540	1,091

Query 25: Word method is better than bigram method.

Keyword: 'LFG'

	terms	comment
Word-based	LFG	Good keyword
Bigram-based	LF, FG	Bad keyword

$$\frac{tf(LFG)}{tf(LF)} = \frac{88}{1127} \qquad \frac{tf(LFG)}{tf(FG)} = \frac{88}{1231}$$

Query4: Bigram method is better than word method.

Keyword:'文書画像理解' (text image understanding)

	terms	comment
Word	文書,画像,理解	Not good keywords
Bigram	文書,画像,理解, 書画, 像理	Not good keywords No sense, but good keywords which represent technical terms

$$\frac{tf(\text{文書画像})}{tf(\text{書画})} = \frac{549}{578} \qquad \frac{tf(\text{画像理解})}{tf(\text{像理})} = \frac{227}{240}$$

Query 26: Both methods are comparable.

Keyword: '語彙機能文法' (Lexical Functional Grammar)

	terms	comment
Word	語彙,機能,文法	Bad keywords for the tech. term
Bigram	語彙,機能,文法, 彙機, 能文	Bad keywords No sense, but good keywords which represent technical terms

$$\frac{tf(\text{語彙機能文法})}{tf(\text{彙機})} = \frac{25}{25}$$

Figure 2: Short words and bigrams are too short to represent technical terms.
A Bigram on word boundary is a good representation of a technical term.

Functional Grammar" in linguistics. The technical term 'LFG' is a good keyword, but the bigram method uses only parts of it; 'LF' and 'FG'. In NTCIR-1 collection, 'LFG' occurs 88 times, but a frequency of 'LF' is 1127 times and a frequency of 'FG' is 1231 times (Figure 2). The term 'LFG' is related to only ten percent of occurrences of 'LF' and 'FG'. The word-based method successes to extract the full term 'LFG' and takes advantage of a full power of this term.

In contrast, the word-based method is clearly worse in queries 4 and 9. Query 4 includes a key-phrase; '文書画像理解' which means "text image understanding" in English. The morphological analysis system used in the word-based method segments the term to three words; '文書' (text), '画像' (image) and '理解' (understanding). These three words are not good keywords, because really large documents include these terms. However, why is the bigram better than three words? In addition to above three words, the bigram-based method uses bigrams on word boundaries; the bigram '書画' extends over '文書' (text) and '画像' (image) and the bigram '像理' extends over '画像' (image) and '理解' (understanding). We found that these meaningless bigrams are very good keywords. The bigram '書画' occurs 573 times and the term '文書画像' (text image) occurs 549 times in the collection. Thus, 90% occurrences of the bigram '書画' is related to '文書画像.' The bigram '書画' is regarded as a good keyword which represents '文書画像.'

The same case is observed in query 26 which includes a technical term '語彙機能文法' ("Lexical Functional Grammar" in English). However, both methods are comparable in the query. The word-based method segmented the term to three words such as '語彙' (Lexical), '機能' (Functional) and '文法' (Grammar) and the bigram uses a very good bigram on word boundaries such as '彙機' which extends over '語彙' and '機能.' All occurrences of '彙機' is a part of the technical term '語彙機能文法' in the collection. The performance suffers from the terms '語彙' and '文法.' Although they are normal good keywords related to the area of natural language processing, they are too general to specify documents related to '語彙機能文法.'

Based on the observation of queries 4 and 26, we will improve the bigram-based method in Section 3.1 and 3.2. Then we will extend bigrams to arbitrary length ngrams to cope with query 25 including a longer technical term than bigrams in Section 3.3.

3. Proposed methods

3.1 Bigram segmentation

There are many selection methods for effective bigrams from a query. In this paper, we focus on the bigram segmentation method which selects non-overlapping bigrams with high tf*idf [7,8]. We think that overlapping bigrams are not only redundant, but also troublesome in some cases. The example in Section 2 showed that bigrams on the word boundaries were good keywords, but words themselves became bad keywords time to time. We'd like to take advantage of only effective bigrams.

The query segmentation method segments a query into a sequence of bigrams and unigrams. Like word segmentation using a dictionary, bigrams and unigrams within a segmentation result do not overlap. Tf*idf measure is used as a criteria for the segmentation. Bigrams or unigrams which have high tf*idf value may be good keywords (key-bigram).

The bigram segmentation method computes just one representation of a query before comparing the query with all documents. Only bigrams in the representation are used to compute the similarity. But, to allow skipped bigrams in the representation, unigrams are used as glue. For example, a phrase '文書の画像' (an image of text) includes a functional word 'の' which means 'of' in English. The best segmentation of this term might be a sequence made up with bigrams and a unigram; '文書' (text) + 'の' (of) + '画像' (image).

Formal definition is the following. The best bigram segmentation, S, maximizes the sum of tf*idf 's of bigrams and unigrams in the segmentation.

$$\hat{S} = \arg\max_{S} \sum_{t \in S} r_{q,t} \cdot r_t \cdot w_t$$

$$r_{q,t} = tf(t,q)$$

$$r_t = 1 + \log tf(t)$$

$$w_t = L(t) \cdot \log(1 + D / df(t))$$

Where $tf(t,q)$ is a term frequency of the term t within the query, q, $tf(t)$ is a total term frequency in the collection and $df(t)$ is a document frequency, D is the number of documents, and $L(t)$ is the length of t. If we don't use the length factor, most units in the segmentation result will be unigrams, because term frequency of unigrams tends to be so larger than those of bigrams. Investigation of the best factor related to length is a future work. In this developing, we used just a length of a term as the factor.

The bigram segmentation is like a stochastic or cost-based morphological analysis for Japanese. The differences are two points; the bigram segmentation uses the text collection itself instead of a dictionary made by hand, and tf*idf instead of probabilities or costs of words.

To compute the best segmentation, the system uses the Viterbi algorithm. All possible segmentation candidates of a query are represented in a lattice made up with unigrams and bigrams at all positions in the query. Figure 3 shows an example of the lattice of the phrase '

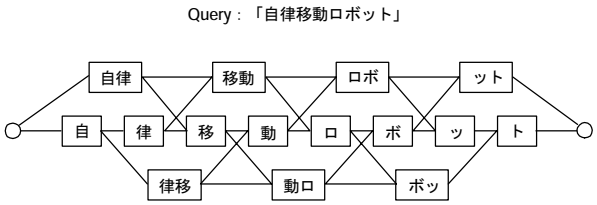

Figure 3: Unigrams and Bigrams Lattice

自律移動ロボット' (autonomous mobile robot). The lattice is made up with 8 unigrams, 7 bigrams and possible connections between them. A path from the left edge to the right edge is corresponding to one segmentation candidate. The tf*idf values for each node are easily computed before the search. Using the Viterbi search technique, the best path which maximizes the sum of tf*idf on the path is computed effectively.

3.2 Adaptive Bigram Segmentation

The bigram segmentation method is intended to extract good bigrams without noisy bigrams. However, this strategy has a risk to miss documents which match only bigrams excluded by the method. If a bigram on a boundary between words is preferred for the method and the words are made up with two characters, bigrams within the words are excluded from the representation of the query. We believe that the bigrams selected by the method is good at high ranks, but some of excluded bigrams may be good at medium ranks. If a query term '文書画像' (a text image), for example, is segmented to '文' + '書画' + '像,' the bigram '書画' is a good keyword for the collection, we believe. However, this segmentation might miss documents including a phrase '文書の画像' (an image of text), because this phrase dose not include the bigram '書画', though both have the same meaning.

To avoid this missing, we define the 'adaptive bigram segmentation' method which computes the best segmentation for each document. For some documents including the term '文書画像', the adaptive segmentation method must segment the key-phrase '文書画像' to '文' + '書画' + '像'. However, for some other documents including the phrase '文書の画像' (an image of text), the method is expected to segment the query to '文書' (text) + '画像' (image), because the bigram on word boundary '書画' is not appear in the documents.

Since the adaptive method computes the best segmentation for each document, it can adopt a real similarity measure between queries and documents as the segmentation criteria. Instead of a total term frequency in the formula for the bigram segmentation method, the adaptive one uses a term frequency within the document.

The similarity measure between a query, q, and a document, d, is defined as the maximum similarity in

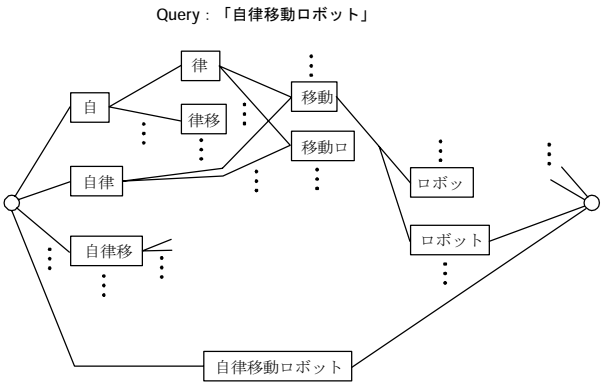

Figure 4 : Ngrams lattice

computed by the possible query segmentations. We referred to Zobel and Moffat paper [9] to determine parameters of the similarity measure.

$$Sim(q,d) = \max_{S} \frac{1}{\sqrt{|d|}} \sum_{t \in S} r_{q,t} \cdot r_{d,t} \cdot w_t$$

The similarity is computed by the Viterbi search on the same lattice used for the bigram segmentation method. The IR system repeats the search on the lattice for each document related to a query.

3.3 Adaptive ngram segmentation

Recall the analysis of query 25 which used the technical term 'LFG.' In this case, both of 'LF' and 'FG' are bad keywords and the bigram framework cannot take advantage of the full technical term. To improve this problem, bigrams are naturally extended to longer ngrams or combination of arbitrary length ngrams such as unigram, bigram, trigram, and so on. The segmentation framework can effectively and automatically select a few good ngrams from many overlapping ngrams.

The Viterbi algorithm is used again on the lattice which includes arbitrary length ngrams in a query as nodes (Figure 4). The best path for a document maximizes the same similarity measure of the adaptive bigram segmenation method, *Sim(q,d)*. The most expensive piece of the computation is the calculation of document frequencies for all ngrams. However, using Suffix Arrays and preprocessing, it is not so hard to compute all df's and total tf's of all substrings in the collection [10,11].

4. Experiments

4.1 Systems and test collection

Specifications and nicknames of the systems used in this section are in Table 1. The test collection NTCIR-1 (preliminary version)[6] is used for evaluation.

Table 1: Summary of the methods described in this paper.

Word-based method:
Word-segmentor: ChaSen[12]
Keywords: noun, verb and unknown words
Similarity measure: $sim_{std}(q,d)$ (better than $sim_L(q,d)$)
Nickname: 'Word'

Bigram-based method:
Keywords: all bigrams in a query.
Similarity measure: $sim_{std}(q,d)$ (better than $sim_L(q,d)$)
Nickname: 'Bigram'

Bigram segmentation method:
Keywords: bigram segmentation of a query
Segmentation criteria: $seg(q)$ (limited to bigram segmentation)
Similarity measure: $sim_{std}(q,d)$
Nickname: 'Bi-Seg.'

Ngram segmentation method:
Keywords: ngram segmentation of a query
Segmentation criteria: $seg(q)$
Similarity measure: $sim_{std}(q,d)$
Nickname: Ngram-Seg.

Adaptive bigram segmentation method:
Keywords: best bigram segmentation for each document.
Segmentation criteria: $sim_L(q,d)$ (limited to bigram segmentation)
Similarity measure: $sim_L(q,d)$
Nickname: Adaptive-Bi-Seg.

Adaptive ngram segmentation method: (used for NTCIR workshop)
Keywords: best ngram segmentation for each document
Segmentation criteria: $sim_L(q,d)$
Similarity measure: $sim_L(q,d)$
Nickname: Adaptive-Ngram-Seg.

$$sim_{std}(q,d) = \frac{1}{\sqrt{|d|}} \sum_{t:L(t)\neq 1} tf(t,q)\cdot(1+\log tf(t,d))\cdot\log\left(1+\frac{D}{df(t)}\right)$$

$$sim_L(q,d) = \frac{1}{\sqrt{|d|}} \sum_{t:L(t)\neq 1} tf(t,q)\cdot(1+\log tf(t,d))\cdot\log\left(1+\frac{D}{df(t)}\right)\cdot L(t)$$

$$seg(q) = \sum_t (1+\log ttf(t))\cdot\log\left(1+\frac{D}{df(t)}\right)\cdot L(t)$$

where q is a query, d is a document, $|d|$ is the length of the document d, $tf(t,q)$ and $tf(t,d)$ are term frequency of the term t within the query q and the document d, D is the number of documents in the collection, $df(t)$ is document frequency of the term t, $ttf(t)$ is total term frequency of the term t in the collection, $L(t)$ is the length of the term t.

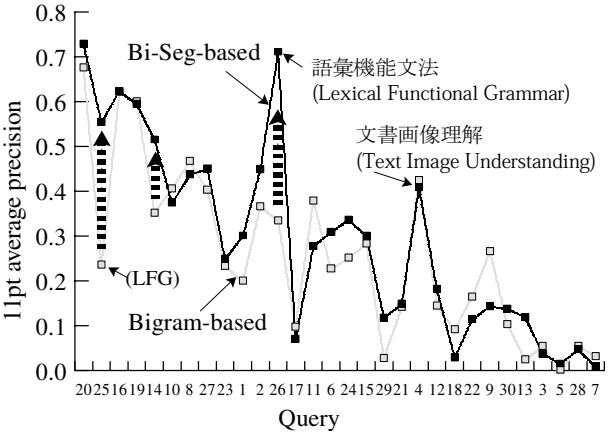

Figure 5: Bi-Seg improves the performance of hard queries for the bigram-based method.

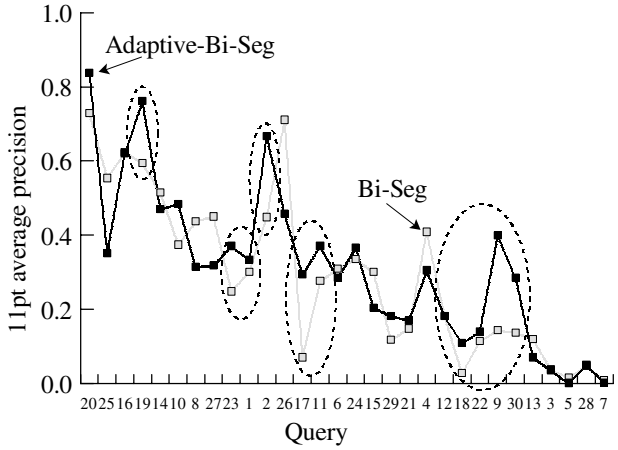

Figure 7: Adapvie-Bi-Seg improves the performance of hard queries for Bi-Seg.

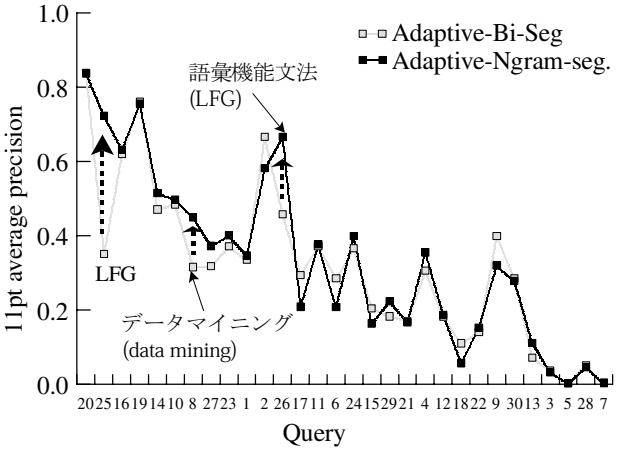

Figure 8: Adaptive-Ngram-Seg improves the performance of hard queries for Adaptive-Bi-Seg.

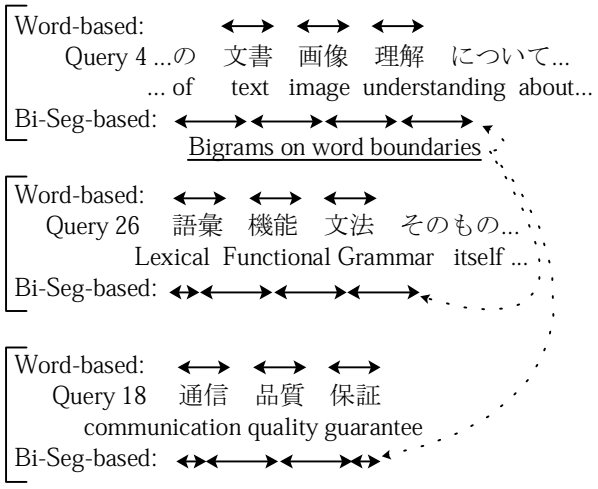

Figure 6: Bi-Seg segments a keyword into bigrams on word boundaries.

4.2 Bigram vs. Bi-Seg.

Figure 5 shows the 11pt. average precision of Bigram and Bi-Seg. systems for each queries. In this figure, the performance for query 26 is greatly improved. Note that Bi-Seg was intended to improve the performance for query 26 which includes '語彙機能文法' (Lexical Functional Grammar). The advantage of the bigram-based method against word-based method in query 4 including '文書画像理解' (text image understanding) is preserved. Up to our expectations, Bi-Seg segmented the phrases in the queries into bigrams on the word boundaries (Figure 6).

However, the performance for some queries (11, 18 and 9) are disappointment. Query 18 includes a technical term '通信品質保証' (communication quality guarantee) which is also segmented into bigrams on the word boundaries which are good keywords (Figure 6). But this technical term is not rigid and is allowed to have many variations in the documents such as '通信の品質の保証' (a guarantee of quality of communication) or '通信の品質を保証する' (to guarantee the quality of communication). In this case, bigrams on the word boundaries miss the documents including the variations of the technical term. We proposed the adaptive bigram segmentation method to avoid those risks.

4.3 Adaptive bigram segmentation

Figure 7 shows the precision graph of Bi-Seg. and Adaptive-Bi-Seg. For queries 11, 18 and 9 in which Bi-Seg. is worse than Bigram (Figure 5), the performance is restored by Adaptive-Bi-Seg. For some of the other queries such as 19 and 2. Adaptive-Bi-Seg. outperforms Bi-Seg. and Word-based method.

However, query 26 including '語彙機能文法' (Lexical Functional Grammar) is hard for Adaptive one. This drop is caused by over-adaptation to irrelevant documents. We cannot expect more improving as long

as using bigrams.

4.4 Adaptive ngram segmentation

Using arbitrary length ngrams, the adaptive segmentation method is also improved for the queries including long technical terms such as 'LFG', 'データマイニング' (data-mining) and '語彙機能文法' (Lexical Functional Grammar) (Figure 8).

Figure 9 shows the 11pt. average precision for each query of the word-based and bigram-based methods and Adaptive-ngram-Seg. This indicates that Adaptive-ngram-Seg. improves the performance in the hard queries for the bigram-based method to the comparable level of the word-based method, and retains the advantage of the bigram-based method against the word-based method, and also improves the comparable queries for Bigram and Word methods. Table 2 summarizes the total 11pt. average precision of all the methods. Segmentation methods are better than the basic method for bigrams. Adaptive methods are also better than the segmentation method.

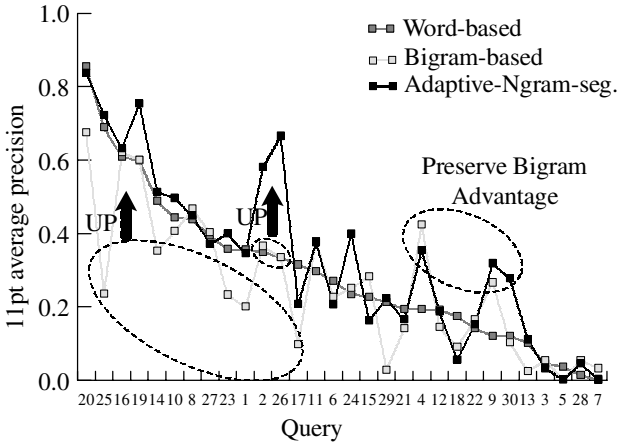

Figure 9: Adaptive-Ngram-Seg. is better than both of Word- and Bigram-based methods.

Query 1: 自律 移動 ロボット
autonomous mobile robot

Ngrams with underline are matched.

in Rank 1 Document: 自律移動ロボット
in Rank 7 Document: 自律移動型マイクロロボット
micro type
in Rank 21Document: 自律型移動ロボット
in Rank 35 Document: 車輪式移動ロボット
wheel type
in Rank 56 Document: 自律分散...による超冗長ロボット
in Rank 59 Document: 自律分散型ロボット
in Rank 127 Document: 自律走行ロボット
moving

Figure 10: Adaptive method changes the segmentation for each document.

Table 2: Ngram is better than bigram.
Adaptive method is better than just one seg.
Combination works better than either by itself.

	all	Just one seg.	Adaptive Seg.
Word	---	0.294	---
Bigram	0.256	0.294	0.315
Ngrams	0.305	0.306	0.336

Figure 10 shows examples of adaptive segmentations of '自律移動ロボット' (autonomous mobile robot) of query 1 for each document. The figure indicates that for the higher ranked document the query is segmented into longer ngrams. Since the document ranked in No.1 includes the whole technical term, the adaptive method does not segment the technical term into words. As the ranks are going down, the matched parts in the documents turn into the variations of the term which have the similar meaning. For example, the document ranked in No.56 includes the long term which specifies the robot in more details.

The adaptive segmenation method can be also extended to preserve word or character order, exactly like the speech recognition. The method is more powerful to take advantage of variations of long technical terms [13].

5. Conclusion

Word segmentation methods such as ChaSen can be viewed as a search over all possible segmentations looking for the path that optimizes an objective function (ex. the minimum connective-cost method with a dictionary). In this work, we considered some other objective functions inspired by research in Information Retrieval. Section 3 described a procedure for segmenting queries into a sequence of ngrams that maximize tf*idf scores along the segmentation path. This procedure will sometimes segment the same string differently depending on the document. We call this adaptative segmentation. The experiments reported in table 2 find that ngrams work better than bigrams, and more interestingly, that adaptative methods work better than non-adaptive methods. The combination of both adaptation and ngrams works better than either by itself. The combination also works better than words.

References

[1] Lawrence Rabiner and Biing-Hwang Juang: Fundamentals of Speech Recognition, PTR Prentice-Hall, 1993.

[2] Joon Ho Lee and Jeong Soo Ahn: "Using n-grams for Korean text retrieval," In proceeding of *SIGIR'96* Zurich, Switzerland, pp.216--224, 1996.

[3] Aitao Chen, Jianzhang He, Liangjie Xu, Fredric C.

Gey, and Jason Meggs: "Chinese text retrieval without using a dictionary," In proceeding of *SIGIR'97,* Philadelphia PA, USA, pp.42-49, 1997.

[4] K. L. Kwok: "Comparing representations in Chinese information retrieval," In proceedings of SIGIR'97, pp.34-41, 1997.

[5] Yasushi Ogawa and Toru Matsuda: "Overlapping statistical word indexing: A new indexing method for Japanese text," In proceeding of *SIGIR'97,* Philadelphia PA, USA, pp.226--234, 1997.

[6] Kageura, K., Teruo Koyama, Masaharu Yoshioka, Atsuhiro Takasu, Toshihiko Nozue, and Keita Tsuji: "NACSIS corpus project for IR and terminological research.", In *Natural Language Processing Pacific Rim Symposium'97*, Phuket, Thailand, pp.493-496, December 2-5, 1997.

[7] Karen Sparck Jones: "Search term relevance weighting given little relevance information," *Journal of Documentation*, Vol.35, No.1, pp.30-48, March 1979.

[8] G. Salton and M.J. McGill: The SMART and SIRE Experimental Retrieval Systems, pp.118-155, NewYork: McGraw-Hill, 1983.

[9] Justin Zobel and Alistair Moffat: "Exploring the similarity space," *SIGIR FORUM*, Vol.32, No.1, pp.18-34, 1998.

[10] Udi Manber and Gene Myers: "Suffix arrays: A new method for on-line string searches," *the first Annual ACM-SIAM Symposium on Discrete Algorithms*, pp.319-327, 1990. URL=http://glimpse.cs.arizona.edu/udi.html

[11] Mikio Yamamoto and Kenneth W. Church: "Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus," In proceeding of *6th Workshop on Very Large Corpora*, Ed. Eugene Charniak, Montreal, pp.28-37, 1998.

[12] Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, Yoshitaka Hirano, Osamu Imaichi, and Tomoaki Imamura: Japanese Morphological analysis System ChaSen Manual, NAIST Technical Report, NAIST-IS-TR97007, February 1997, http: //cactus.aist-nara.ac.jp/ lab/ nlt/ chasen.html.

[13] E. Yamammoto, K. Umemura, T. Ozawa, M. Yamamoto and K.W. Church: "Character based information retrieval using generalized string similarity," IREX workshop, to apear, 1999.