Segmenting Sentences into Linky Strings Using D-bigram Statistics

Shiho Nobesawa
Junya Tsutsumi, Sun Da Jiang, Tomohisa Sano, Kengo Sato
Masakazu Nakanishi

Nakanishi Laboratory, Keio University
3-14-1 Hiyoshi, Kohoku-ku
Yokohama 223 Japan
shiho@nak.math.keio.ac.jp

Abstract

It is obvious that segmentation takes an important role in natural language processing(NLP), especially for the languages whose sentences are not easily separated into morphemes. In this study we propose a method of segmenting a sentence. The system described in this paper does not use any grammatical information or knowledge in processing. Instead, it uses statistical information drawn from non-tagged corpus of the target language. Most of the segmenting systems are to pick out conventional morphemes which is defined for human use. However, we still do not know whether those conventional morphemes are good units for computational processing.

In this paper we explain our system's algorithm and its experimental results on Japanese, though this system is not designed for a particular language.

1 Characteristics of Japanese Text

1.1 Letters in Japanese

Japanese text is composed of four kinds of characters — kanji, hiragana, katakana, and others such as alphabetic characters and numeral characters. Hiragana is used for Japanese words, inflections and function words, while katakana is used for words from foreign languages and for other special purposes.

Table 1 shows examples of rates of those four characters in texts (Teller and Batchelder, 1994). The bus. corpus consists of a set of newspaper articles on business ventures from *Yomiuri*. The ed. corpus contains a series of editorial columns from *Asahi Shinbun*.

Table 1: Character Rates in Japanese Text

	bus.	ed.
size(K chars)	42	275
% hiragana	30.2	58.0
% kanji	47.5	34.6
% katakana	19.3	4.8
% num/alph	2.9	2.6

1.2 Morphemes in Japanese

Segmenting a Japanese text is a difficult task. A phrase "勉強していました (was studying)" can be a single lexical unit or can be separated into as many as six elements (Teller and Batchelder, 1994):

勉強	し	て	い	まし	た
'study'	'do'	particle	progressive	polite	past

Acquiring "morphemes" from Japanese text is not a simple task because of this flexibility.

2 Linky Strings

This paper is on dividing non-separated language sentences into meaningful strings of letters without using any grammar or linguistic knowledge. Instead, this system uses the statistical information between letters to select the best ways to segment sentences in non-separated languages.

It is not very hard to divide a sentence using a certain dictionary for that. The problem is that a 'certain dictionary' is not easily obtainable. There never is a perfect dictionary which holds all the words that exist in the language. Moreover, building a dictionary is very hard work, since there are no perfect automatic dictionary-making systems.

However, machine-readable dictionaries are needed anyway. For this reason, we propose a new method for picking out meaningful strings. Our purpose is not to segment a sentence into conventional morphemes. We introduce a concept for a type of language unit for machine use. We named the unit a 'linky string'. A linky string is a series of letters extracted from a corpus using statistical information only. It is a series of letters which share a strong statistical relationship.

3 Linking Score

3.1 Linking Score

To pick out linky strings, we need to find highly connectable letters in a sentence. We introduce the ***linking score***, which shows the linkability between two neighbor letters in a sentence. This score is estimated using d-bigram statistics.

3.2 D-bigram

The idea of bigrams and trigrams is often used in studies on NLP. N-gram is the information of the association between n certain events. In this study we use the d-bigram data (Tsutsumi et al., 1993), which is a kind of bigram data with the concept of distance between events (Figure 1). D-bigram is equal to bigram when $d = 1$, thus d-bigram data includes the conventional bigram relation.

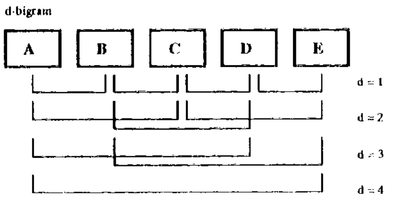

Figure 1: D-bigram

3.3 Calculation

Mutual Information with Distance

Expression (1) is for calculating mutual information between two events(Nobesawa et al., 1994):

$$MI(a_i, b_j, d) = \log_2 \frac{P(a_i, b_j, d)}{P(a_i)P(b_j)} \qquad (1)$$

a_i	:	a letter
$P(a_i)$	:	the possibility the letter a_i appears
$P(a_i, b_j, d)$	:	the possibility a_i and b_j appear together with the distance d in a sentence

The parameter d shows the distance between two events. In Figure 2, the distance between "a" and "pen" is 1, and the distance between "is" and "pen" is 2 as well. Since the event order has a meaning, in this case the distance between "pen" and "a" is defined as -1.

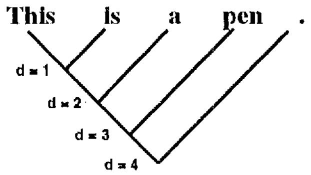

Figure 2: D-bigram Example

As the value of MI gets bigger, the stronger is the association between the two events.

Linking Score

Expression (2) is for calculating the linking score between two letters in a sentence[1].

$$UK[i] = \sum_{d=1}^{d_{max}} \sum_{j=i-(d-1)}^{i} MI(w_j, w_{j+d}, d) \cdot g(d) \qquad (2)$$

d_{max}	:	max distance used
w_i	:	the i-th letter in the sentence w
$g(d)$	:	a certain weight for MI concerning distance between letters

The information between two remote words has less meaning in a sentence when it comes to the semantic analysis(Church and Hanks, 1989). According to the idea we put $g(d)$ in the expression so that nearer pair can be more effective in calculating the score of the sentence.

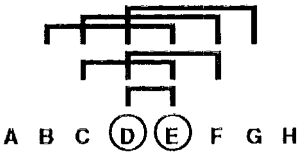

Figure 3: Calculation of Linking Score

A pair of far-away letters do not have strong relation between each other, neither syntactically nor semantically. For this reason we use d_{max}, and in this paper we set the d_{max} value[2] to 5 and 1. When the d_{max} is 1, the MI used in calculation is only bigram data.

[1] We made a Japanese word "有繋" for the word "linky". We used it's pronunciation "UK [ju:kei]" in the expression.

[2] We had experiments for finding a good value for d_{max}.

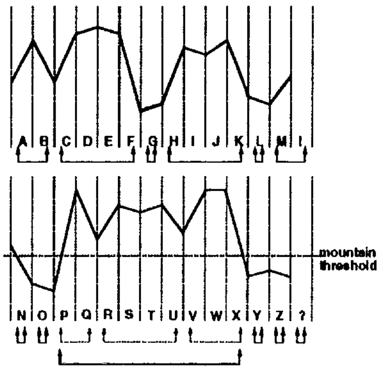

Figure 5: The Score Graph

4 The System *LSS*

4.1 Overview

This system is called *LSS* , a "linky string segmentor". This system takes a corpus made of non-separated sentences as its input and segments it into linky strings using d-bigram statistics.

Figure 4 shows the flow of *LSS* 's processing.

Input sentences to segment.
⇓
Calculate the linking score of
each pair of neighboring letters.
⇓
Check the score graph
to see where to segment.
⇓
pick out each linky string
found in the given corpus.

Figure 4: System Processing Flow

In this paper we used a fixed score for the starting score, so that *LSS* can decide whether the first letter should be a one-letter linky string.

4.2 The Score Graph

What a Score Graph Is

To segment a sentence into statistically-meaningful strings, we use the linking scores to locate boundaries between linking strings. A *score graph* has the letters in a sentence on the x-axis and linking scores on the y-axis (Figure 5). We get one score graph for each sentence. Figure 5 shows two sentences (one above and one below), each of 14 letters (including an exclamation/question mark as the sentence terminator).

When the linking score between a pair of neighboring letters is high, we assume they are part of the same word. When it is low, we assume that the letters, though neighbors, are statistically independent of one another. In a score graph, a series of scores in the shape of mountain (ex.: A-B and C-F part in Figure 5) becomes a linky string, and a valley (ex.: between the letter B and C in Figure 5) is a spot to segment.

Score-Graph Segmenting Algorithm

The system *LSS* finds the valley-points in a sentence and segments the sentence there into strings.

Following is the algorithm to find the segmenting points in a sentence.

1. Do not segment in a mountain.
2. Segment at the valley point.
3. Cut before and after a one-lettered linky string.

One-Lettered Linky String

A one-lettered linky string needs to (a) place at the valley point, and (b) look flat[3] in the score graph. In Figure 5, one-lettered linky strings are G, L, N[4], O, Y, Z and ?.

Mountain Threshold

A linky string takes a mountain shape because of high linking scores. Note that a linky string is not equal to a morpheme in human-handmade grammars. When a certain pair of morphemes occurs in a corpus very often, the system recognizes the pair's high linking score and puts them together into one linky string. For example, "ブッシュ大統領 (President Bush)" is often treated as a linky string, since "ブッシュ(Bush)" and "大統領 (president)" appear next to each other very frequently.

The mountains of letters are not always simple hat-shaped; most of time they have other smaller mountains in them. This means that there can be shorter strings in one linky string. In one linky string "ブッシュ大統領 (President Bush)", there must be two smaller mountains, just like H-I and J-K in the mountain H-K in Figure 5. To control the size of linky strings we introduce a *mountain threshold*, which is shown in the sentence below in Figure 5. When the score of a valley point is higher than the mountain threshold, the system judges the point is not a segmenting spot. In this paper the mountain threshold value is 5.0.

[3] We use a constant value as a threshold.

[4] N is a special one-lettered linky string which places at the beginning of a sentence.

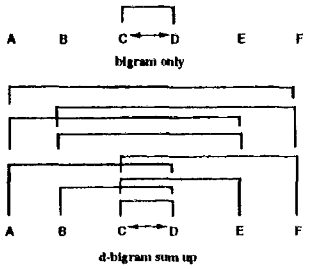

Figure 6: The Difference between D-bigram and Bigram

4.3 Corpus

ISS accepts all the non-separated sentences with little preparation. All we need is a set of certain amount of the target-language corpus for training.

In this paper we show the experimental results on Japanese. The corpus prepared for this paper is of Asahi Shinbun Newspaper.

5 RESULTS

5.1 Experimental Results

Experiment Condition

ISS takes a set of non-separated sentences as its input and segments them into linky strings. For the test corpus we chose sentences at random from the training corpus.

Table 2: Training Corpus Condition

language:	Japanese
form:	non-separated kanji-kana mixed sentences
corpus:	Asahi Shinbun Newspaper
# of sentences for training corpus:	7,502
# of sentences for test corpus:	302

To see the efficacy of d-bigram, we compare the experimental results of two data: d-bigram data and bigram data.

Experimental Results

As shown in Table 3, with d-bigram information only 7.39% of the segment spots are over-segmented.

Table 3: Output

	d-bigram $d_{max} = 5$	bigram
# of input sentences	302	302
# of linky strings	6,145	7,098
# of linky strings per sentence	20.35	23.50
# of over-segmented spots	454	689
over-segmented spots	7.39%	9.71%

Table 3 shows that a sentence gets separated into 20–25 linky strings on average[5]. And in one sentence there are only one or two spots on average which break a morpheme into meaningless strings. With no linguistic knowledge, this can be said to be quite a good result.

It is hard to check whether an extracted linky string is a right one, however, it is not that difficult to find over-segmented strings, for a linky string needs to hold the meaning. We check those over-segmented linky strings according to a dictionary, *Iwanami Kokugo Jiten*.

Table 4 shows the numbers of over-segmented spots. The figure is the number of over-segmented spots, not the number of morphemes over-segmented[6]. In Table 4 A and B are neighboring letters in a sentence which are forced to separate. The row "kanji hiragana" stands fór over-segmented spots between a kanji letter and a hiragana letter.

Table 4: Over-Segmented Morphemes by Character Types and Segmentation Methods

A	B	d-bigram $d_{max} = 5$	bigram
kanji	kanji	59	65
kanji	hiragana	29	43
hiragana	kanji	18	22
hiragana	hiragana	333	507
katakana	katakana	15	52
	total	454	689

The ratio of over-segmented morphemes for each part of speech is shown in Table 5. 'K' stands for kanji, 'h' is for hiragana and 'k' is for katakana.

There was no missegmentation between katakana and other character types. There also was not any

[5] The range of numbers of linky strings found in a sentence is 5–60 with d-bigram and 6–66 with bigram.

[6] Thus a morpheme gets counted twice when it is divided into three strings.

Table 5: Over-Segmented Morphemes in Output with D-bigram

A B	K K	K h	h K	h h	k k	 total
noun	49	19	6	49	6	129
proper noun	5				8	13
pronoun				16		16
verb	1	3	12	84		100
aux. verb				60		60
adjective		4		12		16
adj. verb	4			13		17
adverb		1		53	1	55
rentai-shi				11		11
conjunction				7		7
funcion word				15		15
suffix		1		4		5
compound word		1		15		16
total	59	29	18	333	15	454

missegmentation concerning alphabets, numeral characters and other symbols.

5.2 A Linky String

Characteristics of Linky Strings

Linky strings in Japanese are not equal to conventional morphemes in Japanese. As discussed in section 1.2, it is not easy to decide an absolutely correct segmenting spot in a Japanese sentence. That is one of the reasons that we decided to extract linky strings, instead of conventional morphemes. However, if those linky strings do not keep the meanings, it is useless.

The result shows the linking score works well enough not to segment senteces too much (Table 3). That is, we succeeded in extracting meaningful strings using only statistical information. Figure 7 shows some examples of extracted linky strings.

銀行	bank(s)	meaningful
に移行	move/shift to	meaningful
の行動	action of	meaningful
を行った	did	meaningful
(?) 行 (?)		over-segmented
(年) 中行事		over-segmented

Figure 7: Examples of Linky Strings (1)

Sometimes *LSS* extracts strings that look too long (Figure 8). This is not a bad result, though. When a linky string contains several morphemes in it, it is something like picking out idioms. A linky string with several morphemes may be a compound word, or an idiom, or a fixed locution.

手を貸す	help
ロンドン・サミット	London Summit
核拡散防止条約	nuclear non-proliferation treaty
１７世紀の末	at the end of 17th century
ＪＲ京都駅	Japan Railway Kyoro Station

Figure 8: Examples of Linky Strings (2)

The Concept of the Linky Strings

Grammar-based NLP systems generally specify a target language. On the other hand statistically-based approachs do not need rules or knowledge. This makes a statistically-based approach suitable to multilingual processing.

LSS is not only for Japanese. With a corpus of non-separated sentences of any language, *LSS* can perform the same kind of segmentation.

To deal with natural languages most systems use conventional morphemes or words as their processing units. That is, most systems need to recognize morphemes or words in sentences, and they need to make up a fairly-good morphological analysis before the main processing. We have been working for processing natural languages in linguistic ways, though we do not know whether it is a right way in computational linguistics. A linky string is extracted only with statistical information, using no grammars nor linguistic knowledge. The system does not need to behave like a native speaker of the target language; all it has to do is check statistical information, which is what computers are good at. We expect that linky strings can be a key to solve problems of NLP.

Compound Words

The results show that the system has 7.39% incorrect segmentation. This result is based on a Japanese dictionary, and when a morpheme listed in the dictionary gets separated, we count it as over-segmented. However, a dictionary often holds compound words. That is, some number of the segmented spots which we have counted as "over-segmented" ones are not really over-segmented. From this point of view, the percentage of over-segmentation is actually even lower.

Inflections

Verbs, adjectives, adverbs and auxiliary verbs are inflected in Japanese. In the experimental result, 89.7% (with d-bigram data) of over-segmented spots between kanji and hiragana occurs in inflective morphemes. We decided correct segmenting spots for inflective morphemes according to a Japanese dictionary. According to statistical information, segmenting method for inflective morphemes is different from grammatical one. So most of the over-segmented spots can be treated as correct segmenting spots according to statistical information.

5.3 D-bigram Statistics

According to Table 3, it seems that using the bigram method the output is apt to be more segmented than with the d-bigram method.

This happens because bigram cannot pick out long strings. Bigram does not hold information between remote (actually more than one letter away) letters. That makes long strings of letters easily segmented. When *LSS* checks a three-letter morpheme ABC, with bigram data it can see the string only as A-B and B-C. If those strings AB and BC do not appear often, the linking scores get low and *LSS* decides to segment between A-B and B-C. However, with d-bigram data *LSS* can get the information between A and C as well, that helps to recognize that A, B and C often come out together. This happens frequently between two katakana letters (Table 4), because of the usage of katakana letters in Japanese.

This does not mean that with d-bigram method sentences are less likely to be segmented. As shown in Figure 9, the distribution is not so different between two methods. The x-axis shows the numbers of linky strings in sentences and the y-axis shows the number of sentences with x linky strings.

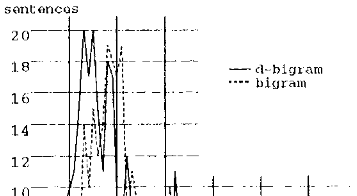

Figure 9: The Number of Strings in Output Sentences

According to Figure 9, the distributions of sentences are not so different between the method with d-bigram and the one with bigram.

6 Conclusion

This paper shows that this automatic segmenting system *LSS* is quite efficient for segmentation of non-separated language sentences. *LSS* does not use any grammatical information to divide input sentences into linky strings, that is, a new unit for NLP. According to the results of the experiments, *LSS* can segment almost all the sentences 'correctly', with strings keeping their meanings. This remarkable result of a statistic-based system *LSS* shows that d-bigram statistical information can be a key to meaningful-string extracting.

This result also shows that the concept of linky strings is an interesting concept for NLP. We expect that this linky string can be a unit for machine translation systems or key word/phrase extraction systems, and other NLP systems.

References

[1] Tsutsumi, J., Nitta, T., Ono, K. and Nobesawa, S. A Multi-Lingual Translation System Based on A Statistical Model(written in Japanese). *JSAI Technical report, SIG-PPAI-9302-2*, pages 7–12, 1993.

[2] Nobesawa, S., Tsutsumi, J., Sun D. J., Sano, T., Sato K. and Nakanishi, M. Automatic Extraction of Linky Strings in Natural Languages (written in Japanese). *2nd Annual Meeting of the ANLP (NLP96)*, pages 181–184, 1996.

[3] Nobesawa, S., Tsutsumi, J., Nitta, T., Ono, K., Sun, D. J. and Nakanishi, M. Segmeting a Japanese Sentence into Morphemes Using Statistical Information between Words. *Coling-94*, pages 227–233, 1994.

[4] Teller V. and Batchelder E. O. A Probabilisitic Algorithm for Segmenting Non-Kanji Japanese Strings. *AAAI*, 1994.

[5] Brown, P., Cocke, J., Pietra, S. D., Pietra, V. D., Jelinek, F., Mercer, R. and Roossin, A. A Statistical Approach to Language Translation. *Coling-88*, pages 71–76, 1988.

[6] Church, K. and Hanks, P. Word Association Norms, Mutual Information, and Lexicography. *In Proceedings of the 27th Annual Conference of the association of Computational Linguistics*, 1989.

[7] Nishio, M., Iwabuchi, E. and Mizutani, S. *Japanese-Japanese Dictionary The 3rd Edition.* Iwanami Shoten, 1985.