Retrieving Collocations by Co-occurrences and Word Order Constraints

Sayori Shimohata, Toshiyuki Sugio and **Junji Nagata**
Kansai Laboratory, Research & Development Group
Oki Electric Industry Co., Ltd.
Crystal Tower 1-2-27, Shiromi,
Chuo-ku, Osaka, 540, Japan
{sayori,sugio,nagata}@kansai.oki.co.jp

Abstract

In this paper, we describe a method for automatically retrieving collocations from large text corpora. This method retrieve collocations in the following stages: 1) extracting strings of characters as units of collocations 2) extracting recurrent combinations of strings in accordance with their word order in a corpus as collocations. Through the method, various range of collocations, especially domain specific collocations, are retrieved. The method is practical because it uses plain texts without any information dependent on a language such as lexical knowledge and parts of speech.

1 Introduction

A collocation is a recurrent combination of words, ranging from word level to sentence level. In this paper, we classify collocations into two types according to their structures. One is an uninterrupted collocation which consists of a sequence of words, the other is an interrupted collocation which consists of words containing one or several gaps filled in by substitutable words or phrases which belong to the same category.

The features of collocations are defined as follows:

- collocations are recurrent
- collocations consist of one or several lexical units
- order of units are rigid in a collocation.

For language processing such as machine translation, a knowledge of domain specific collocations is indispensable because what collocations mean are different from their literal meaning and the usage and meaning of a collocation is totally dependent on each domain. In addition, new collocations are produced one after another and most of them are technical jargons.

There has been a growing interest in corpus-based approaches which retrieve collocations from large corpora (Nagao and Mori, 1994), (Ikehara et al., 1996) (Kupiec, 1993), (Fung, 1995), (Kitamura and Matsumoto, 1996), (Smadja, 1993), (Smadja et al., 1996), (Haruno et al., 1996). Although these approaches achieved good results for the task considered, most of them aim to extract fixed collocations, mainly noun phrases, and require the information which is dependent on each language such as dictionaries and parts of speech. From a practical point of view, however, a more robust and flexible approach is desirable.

We propose a method to retrieve interrupted and uninterrupted collocations by the frequencies of co-occurrences and word order constraints from a monolingual corpus. The method comprises two stages: the first stage extracts sequences of words (or characters) [1] from a corpus as units of collocations and the second stage extracts recurrent combinations of units and constructs collocations by arranging them in accordance with word order in the corpus.

2 Algorithm

2.1 Extracting units of collocation

(Nagao and Mori, 1994) developed a method to calculate the frequencies of strings composed of n characters(n grams). Since this method generates all n-character strings appeared in a text, the output contains a lot of fragments and useless expressions. For example, even if "local", "area", and "network" always appear as the substrings of "a local area network" in a corpus, this method generates redundant strings such as "a local", "a local area" and "area network".

To filter out the fragments, we measure the distribution of adjacent words preceding and following

[1] A word is recognized as a minimum unit in such a language as English where writespace is used to delimit words, while a character is recognized as that in such languages as Japanese and Chinese which have no word delimiters. Although the method described in this paper is applicable to either kinds of languages, we have taken English as an example.

the strings using entropy threshold. This is based on the idea that adjacent words will be widely distributed if the string is meaningful, and they will be localized if the string is a substring of a meaningful string. Taking the example mentioned above, the words which follow "a local area" are practically identified as "network" because "a local area" is a substring of "a local area network" in the corpus. On the contrary, the words which follow "a local area network" are hardly identified because "a local area network" is a unit of expression and innumerable words are possible to follow the string. It means that the distribution of adjacent words is effective to judge whether the string is an appropriate unit or not.

We introduce entropy value, which is a measure of disorder. Let the string be str, the adjacent words $w_1 ... w_n$, and the frequency of str $freq(str)$. The probability of each possible adjacent word $p(w_i)$ is then:

$$p(w_i) = \frac{freq(w_i)}{freq(str)} \quad (1)$$

At that time, the entropy of str $H(str)$ is defined as:

$$H(str) = \sum_{i=1}^{n} -p(w_i) \log p(w_i) \quad (2)$$

$H(str)$ takes the highest value if $n = freq(str)$ and $p(w_i) = \frac{1}{n}$ for all w_i, and it takes the lowest value 0 if $n = 1$ and $p(w_i) = 1$. Calculating the entropy of both sides of the string, we adopt the lower one as the entropy of the string. Str is accepted only if the following inequation is satisfied:

$$H(str) \geq T_{entropy} \quad (3)$$

Fragmental strings such as "a local" and "area network" are filtered out with these procedures because their entropy values are expected to be small. Most of the strings extracted in this stage are meaningful units such as compound words, prepositional phrases, and idiomatic expressions. These strings are uninterrupted collocations of themselves while they are used in the next stage to construct collocations. This method is useful for the languages without word delimiters, and for the other languages as well.

2.2 Extracting collocations

By the use of each string derived in the previous stage, this stage extracts strings which frequently co-occur with the string and constructs them as a collocation. It is based on the idea that there is a string which is used to induce a collocation. We call this string "a key string", hereafter. The followings are the procedures to retrieve a collocation:

1. Take a key string str_k from the strings $str_i (i = 1...n)$, and retrieve sentences containing str_k from the corpus.
2. Examine how often each possible combinations of str_k and str_i co-occurs, and extract str_i if the frequency exceeds a given threshold T_{freq}.
3. Examine every two strings str_i and str_j and refine them by the following steps alternately:
   - Combine str_i and str_j when they overlap or adjoin each other and the following inequation is satisfied:
   $$\frac{freq(str_i, str_j)}{freq(str_i)} \geq T_{ratio} \quad (4)$$
   - Filter out str_i if str_j subsumes str_i and the following inequation is satisfied:
   $$\frac{freq(str_j)}{freq(srt_i)} \geq T_{ratio} \quad (5)$$
4. Construct a collocation by arranging the strings str_i in accordance with the word order in the corpus.

The second step and the third step narrow down the strings to the units of collocation. Through these steps, only the strings which significantly co-occur with the key string str_k are extracted.

The second step eliminates the strings that are not frequent enough. Consider the example of Figure 1. This is a list of sentences containing the key string "Refer to" retrieved and each underlined string corresponds to a string str_i. Assuming the frequency threshold T_{freq} as 2, the strings which co-occur with str_k more than twice are extracted in the second step. Table 1 shows the result of this step. Although it is very simple technique, almost all the useless strings are excluded through this step.

str_i	$freq(str_k, str_i)$
the	4
manual	4
for specific instructions	3
on	2

Table 1: Result of the second step

The third step reorganizes the strings to be optimum units in the specific context. This is based on the idea that a longer string is more significant as a unit of collocations if it is frequent enough. Assuming that the threshold T_{ratio} is 0.75, first, a string "manual for specific instructions" is produced as the inequation (4) is satisfied. Next, "manual" and "for specific instructions" are deleted as the inequation (5) is satisfied. This process is repeated until no string satisfies the inequations. Table 2 shows a result of this step.

The fourth step constructs a collocation by arranging the strings in accordance with the word order in the sentences retrieved in the first step. Taking str_i in order of frequency, this step determines

Refer to the appropriate manual for instructions on...
Refer to the manual for specific instructions.
Refer to the installation manual for specific instructions for ...
Refer to the manual for specific instructions on ...

Figure 1: Sentences containing "Refer to"

str_i	$freq(str_k, str_i)$
the	4
manual for specific instructions	3
on	2

Table 2: Result of the third step

where str_i is placed in a collocation. In this example, the position of "the" is examined first. According to the sentences shown in Figure 1, "the" is always placed next to "Refer to". Then its position is determined to follow "Refer to". Next, the position of "manual for specific instructions" is examined and it is determined to follow a gap placed after "Refer to the". Finally, the following collocation is produced:

"Refer to the ... manual for specific instructions on ..."

The broken lines in the collocation indicates the gaps where any substitutable words or phrases can be filled in. In the example, "appropriate" or "installation" is filled in the first gap.

Thus, we retrieve an arbitrary length of interrupted or uninterrupted collocation induced by the key string. This procedure is performed for each string obtained in the previous stage. By changing the threshold, various levels of collocations are retrieved.

3 Evaluation

We performed an experiment for evaluating the algorithm. The corpus used in the experiment is a computer manual written in English comprising 1,311,522 words (in 120,240 sentences).

In the first stage of this method, 167,387 strings are produced. Among them, 650, 1950, 6774 strings are extracted over the entropy threshold 2, 1.5, 1 respectively. For 650 strings whose entropy is greater than 2, 162 strings (24.9%) are complete sentences, 297 strings (45.7%) are regarded as grammatically appropriate units, and 114 strings (17.5%) are regarded as meaningful units even though they are not grammatical. This told us that the precision of the first stage is 88.1%.

Table 3 shows top 20 strings in order of entropy value. They are quite representative of the given domain. Most of them are technical jargons related to computers and typical expressions used in manual descriptions although they vary in their constructions. It is interesting to note that the strings which do not belong to the grammatical units also take high entropy value. Some of them contain punctuation, and some of them terminate in articles. Punctuation marks and function words in the strings are useful to recognize how the strings are used in a corpus.

Table 4 illustrates how the entropy is changed with the change of string length. The third column in the table shows the kinds of adjacent words which follow the strings. The table shows that the ungrammatical strings such as "For more information on" and "For more information, refer to" act more cohesively than the grammatical string "For more information" in the corpus. Actually, the former strings are more useful to construct collocations in the second stage.

In the second stage, we extracted collocations from 411 key strings retrieved in the first stage (297 grammatical units and 114 meaningful units). Necessary thresholds are given by the following set of equations:

$$\left\{ \begin{array}{l} T_{entropy} = 1 \\ T_{freq} = \frac{freq(str_i)}{freq(str_k)} \times 0.1 \\ T_{ratio} = 0.8 \end{array} \right\}$$

As a result, 269 combinations of units are retrieved as collocations. Note that collocations are not generated from all the key strings because some of them are uninterrupted collocations in themselves like No. 2 in Table 3. Evaluation is done by human check and 180 collocations are regarded as meaningful. The precision is 43.8% when the number of meaningful collocation is divided by the number of the key strings and 66.9% when it is divided by the number of the collocations retrieved in the second stage[2].

Table 5 shows the collocations extracted with the underlined key strings. The table indicates that arbitrary length of collocations, which are frequently used in computer manuals, are retrieved through the method. As the method focuses on the co-occurrence of strings, most of the collocations are specific to the given domain. Common collocations are tend to be ignored because they are not used repeatedly in a single text. It is not a serious problem,

[2]Usually the latter ratio is adopted as *precision*.

however, because common collocations are limited in number and we can efficiently obtain them from dictionaries or by human reflection.

No. 7 and 8 in Table 5 are the examples of invalid collocations. They contain unnecessary strings such as "to a" and ", the" in them. The majority of invalid collocations are of this type. One possible solution is to eliminate unnecessary strings at the second stage. Most of the unnecessary strings consist of only punctuation marks and function words. Therefore, by filtering out these strings, invalid collocations produced by the method should be reduced.

Figure 2 summarizes the result of the evaluation. In the experiment, 573 strings are retrieved as appropriate units of collocations and 180 combinations of units are retrieved as appropriate collocations. Precision is 88.1% in the first stage, and 66.9% in the second stage.

<table>
<tr><th>1st stage</th><th>2nd stage</th></tr>
<tr><td>CS=162(24.9%)</td><td>-</td></tr>
<tr><td>GU=297(45.7%)</td><td rowspan="2">MC=180(43.8%)
F=89(21.7%)
NC=142(34.5%)</td></tr>
<tr><td>MU=114(17.5%)</td></tr>
<tr><td>F=77(11.9%)</td><td>-</td></tr>
</table>

CS: complete sentences
GU: grammatical units
MU: meaningful units
MC: meaningful collocations
F: fragments
NC: not captured

Figure 2: Summary of evaluation

Although evaluation of retrieval systems is usually performed with precision and recall, we cannot examine recall rate in the experiment. It is difficult to recognize how many collocations are in a corpus because the measure differs largely dependent on the domain or the application considered. As an alternative way to evaluate the algorithm, we are planning to apply the collocations retrieved to a machine translation system and evaluate how they contribute to the quality of translation.

4 Related work

Algorithms for retrieving collocations has been described (Smadja, 1993) (Haruno et al., 1996).

(Smadja, 1993) proposed a method to retrieve collocations by combining bigrams whose co-occurrences are greater than a given threshold[3]. In their approach, the bigrams are valid only when there are fewer than five words between them. This is based on the assumption that "most of the lexical relations involving a word w can be retrieved by examining the neighborhood of w wherever it occurs, within a span of five (-5 and +5 around w) words." While the assumption is reasonable for some languages such as English, it cannot be applied to all the languages, especially to the languages without word delimiters.

(Haruno et al., 1996) constructed collocations by combining a couple of strings[4] of high mutual information iteratively. But the mutual information is estimated inadequately lower when the cohesiveness between two strings is greatly different. Take "in spite (of)", for example. Despite the fact that "spite" is frequently used with "in", mutual information between "in" and "spite" is small because "in" is used in various ways. Thus, there is the possibility that the method misses significant collocations even though one of the strings have strong cohesiveness.

In contrast to these methods, our method focuses on the distribution of adjacent words (or characters) when retrieving units of collocation and the co-occurrence frequencies and word order between a key string and other strings when retrieving collocations. Through the method, various kinds of collocations induced by key strings are retrieved regardless of the number of units or the distance between units in a collocation. Another distinction is that our method does not require any lexical knowledge or language dependent information such as part of speech. Owing to this, the method have good applicability to many languages.

5 Conclusion

In this paper, we described a robust and practical method for retrieving collocations by the co-occurrence of strings and word order constraints. Through the method, various range of collocations which are frequently used in a specific domain are retrieved automatically. This method is applicable to various languages because it uses a plain textual corpus and requires only the general information appeared in the corpus. Although the collocations retrieved by the method are monolingual and they are not available to the machine application for the present, the results will be extensible in various ways. We plan to compile a knowledge of bilingual collocations by incorporating the method with conventional bilingual approaches.

[3]This approach is similar to the process of the string refinement described in this paper.

[4]They call the strings word chunks.

No.	*str*	$H(str)$	$freq(str)$
1	the current functional area	3.8	45
2	Before you install this device :	3.78	44
3	This could introduce data corruption .	3.37	29
4	All rights are reserved .	3.37	29
5	Note that the	2.93	53
6	, such as	2.91	87
7	Information on minor numbers is in	2.45	20
8	, for example ,	2.44	23
9	The default is	2.44	52
10	, you can use the	2.26	25
11	to see if the	2.2	24
12	stands for	2.15	30
13	system accounting :	2.14	48
14	These are	2.12	37
15	allocation policy	2.1	21
16	For example , the	2.1	97
17	For more information on	2.1	96
18	permission bits	2.07	26
19	By default , the	2.06	32
20	The syntax for	2.03	57

Table 3: Top 20 strings extracted at the first stage

str	$H(str)$	n	$freq(str)$
For more	0.13	7	200
For more information	0.33	3	168
For more information ,	0.21	4	46
For more information , see	1.03	8	25
For more information , refer to	1.17	6	15
For more information on	2.1	56	96
For more information about	1.69	21	35

Table 4: Strings including "For more"

No.	collocation
1	For more information on ..., refer to the ... manual.
2	You can use the ... to help you.
3	The syntax for .... is : ...
4	output from the execution of ... commands.
5	..., use the ... command with the ... option
6	... have a special meaning in this manual.
7	... to a (such as ..., and ...).
8	... if the system ...or a ...for a...,the..

Table 5: Examples of collocations extracted at the second stage

References

Pascale Fung. 1995. Compiling bilingual lexicon entries from a non-parallel English-Chinese corpus. In *Proceedings of the 3rd Workshop on Very Large Corpora*, pages 173–183.

Masahiko Haruno, Satoru Ikehara, and Takefumi Yamazaki. 1996. Learning Bilingual Collocations by Word-Level Sorting. In *Proceedings of the 16th COLING*, pages 525–530.

Satoru Ikehara, Satoshi Shirai, and Hajime Uchino. 1996. A statistical method for extracting uninterrupted and interrupted collocations from very large corpora. In *Proceedings of the 16th COLING*. pages 574–579.

Mihoko Kitamura and Yuji Matsumoto. 1996. Automatic extraction of word sequence correspondences in parallel corpora. In *Proceedings of the 4th Workshop on Very Large Corpora*, pages 79–87.

Julian Kupiec. 1993. An algorithm for finding noun phrase correspondences in bilingual corpora. In *Proceedings of the 31th Annual Meeting of ACL*, pages 17–22.

Makoto Nagao and Shinsuke Mori. 1994. New Method of n-gram statistics for large number of n and automatic extranction of words and phrases from large text data of Japanese. In *Proceedings of the 15th COLING*, pages 611–615.

Frank Smadja. 1993. Retrieving collocations from text: Xtract. In *Computational Linguistics, 19(1)*, pages 143–177.

Frank Smadja, Kathleen MaKeown, and Vasileios Hatzivassiloglou. 1996. Translating collocations for bilingual lexicons: A statistical approach. In *Computational Linguistics, 22(1)*, pages 1–38.