Experimental Evaluation of Ranking and Selection Methods in Term Extraction

Appeared in "Recent Advances in Computational Terminology", D. Bourigault, C. Jacquemin, M.-C. L'Homme (editors), pp 303 -- 325 , John Benjamins, 2000

Hiroshi Nakagawa

Information Technology Center, The University of Tokyo

7-3-1 Hongo, Bunkyo, Tokyo, 113-0033, Japan

An automatic term extraction system consists of a term candidate extraction subsystem, a ranking subsystem and a selection subsystem. In this paper, we experimentally evaluate two ranking methods and two selection methods. As for ranking, a dichotomy of unithood and termhood is a key notion. We evaluate these two notions experimentally by comparing *Imp* based ranking method that is based directly on termhood and C-value based method that is indirectly based on both termhood and unithood. As for selection, we compare the simple threshold method with the window method that we propose. We did the experimental evaluation with several Japanese technical manuals. The result does not show much difference in recall and precision. The small difference between the extracted terms by these two ranking methods depends upon their ranking mechanism *per se*.

1. Introduction

As widely known, automatic term extraction is definitely useful in various areas including (1) Automatic index extraction from a volume of text, (2) Terminology extraction from one academic field, and (3) Keywords extraction from documents for IR purposes. Especially (1) and (2) have so far been done manually and cost too much. Therefore, an automatic term extraction technology would be great help for these purposes. Kageura and Umino (1996:259-289) refer to two essential aspects of the nature of terms, namely unithood and termhood.

Unithood refers to the degree of strength or stability of syntagmatic combinations or collocations. For instance, a word has very solid unithood. Other linguistic units having strong unithood are compound words, collocations, and so forth.

Termhood refers to the degree that a linguistic unit is related to domain-specific concepts. Termhood is usually calculated based on term frequency and bias of frequency (so called Inverse Document Frequency). Even though these calculations give a good approximation of termhood, still they do not directly reflect termhood because these calculations are based on superficial statistics.

According to these two aspects, we have a dichotomy of term extraction methods, namely term extraction based on unithood and that based on termhood. Obviously, terms that have high termhood should be extracted as terms. However, to directly measure termhood of the given term candidate is extremely difficult because only the writer of a document knows which terms are important terms. Many researchers have tried to work out the way to approximate termhood by some score that is often calculated based on unithood so far. Therefore, the question we would like to ask is how directly the given extracting method measures termhood even though it is based on unithood. The accompanying question is what characteristics the terms extracted by each method have. In fact, they are tough questions to answer theoretically. The best thing we can do at this moment is to compare experimentally the performance of several term extraction methods. Since it is still difficult to compare many methods, in this paper, we compare only two methods: C-value based method (Frantzi and Ananiadou 1996:41-46) and *Imp* based method (Nakagawa 1997:598-611).

2. Overview of a Term Extraction System

A term extraction system, in general, consists of three subsystems, namely 1) candidate extraction, 2) ranking, and 3) selection, as shown in Figure 1.

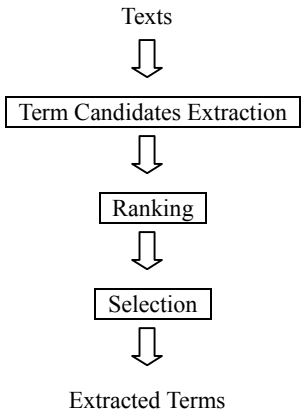


Figure 1: Structure of Term Extraction System

In the following, we sketch each of these three subsystems along with the previous works.

Term Candidates Extraction subsystem

There are two major types of term candidates in terms of linguistic structure. One is an N-gram of characters. The other is a word. Much work has been done on character based N-gram, especially in some Asian languages like Japanese (Fujii and Croft 1993:237-246) and Chinese (Lam et al. 1997:68-80). Since all of these aim at extracting terms for information retrieval, character based N-grams give us enough quality as keywords for IR. However, for non-IR purposes like (1) and (2) above, character based N-grams are not suitable because back of the book indexes or terminologies of one academic field are not superficial sequences of characters but are words bearing semantically coherent information. Therefore, in this paper, we concentrate on terms based on words.

Term candidates that consist of words are nouns or compound nouns. To extract promising term candidates of compound noun and at the same time to exclude undesirable strings such as *is a* or *of the*, the most frequently used method is to filter out the words being members of the stop-list. In these days, more complex structures like noun phrases, collocations consisting of nouns, verbs, prepositions, determiners, and so on, become focused on (Smadja and McKeown 1990:252-259; Frantzi and Ananiadou 1996:41-46, Zhai and Evans 1996:17-23; Hisamitsu and Nitta 1996:550-555, Shimohata et al. 1997:476-481). All of these are good term candidates in a document or a specific domain because all of them have a strong unithood. Needless to say, but as for complex terms like compound words or collocations, we have the following basic assumption:

Assumption *Complex terms are to be made of existing simple terms.*

A structure of complex term is another important factor for automatic term extraction. It is expressed syntactically or semantically. As a syntactic structure, dependency structures that are the results of noun phrase parsing are focused on in many works. Of course, we need heuristics or statistics to select plausible dependency structures (Zhai and Evans 1996:17-23).

Since we focus on these complex structures, the first thing to extract term candidates is morphological analysis including part of speech (POS) tagging. In English, POS tagging has been one of the main issues in natural language processing, i.e. (Brill 1994a:722-727), and recently high quality POS taggers such as (Brill 1994b) are available. In Japanese that is an agglutinative language, morphological analysis segments out words from a sentence, and does POS tagging, too (Matsumoto et al. 1996). After POS tagging, the complex structure mentioned above is extracted as a term candidate. The previous works proposed many promising ways for this type of term candidate extraction. Zhai and Evans (1996:17-23) focus on noun phrases. Ananiadou (1994:1034-1038) proposes the way to extract word compounds as terms. Hisamitsu and Nitta (1996:550-555) and Nakagawa (1997:598-611) concentrate their efforts on

compound nouns. Smadja and McKeown (1990:252-259), Daille et al. (1994:515-521), Frantzi and Ananiadou (1996:41-46) and Shimohata et al. (1997:476-481) try to treat more general structures like collocations.

Ranking subsystem

In order to extract domain specific terms from term candidates extracted in Term Candidates Extraction subsystem, we have to rank them according to their termhood. This ranking has been developed as keyword weighting like *tf*・*idf* which is widely used in IR. As written in (Kageura and Umino 1996:259-289), the frequency information about a word, like *tf*・*idf* , is an approximation of termhood. Obviously, a notion of termhood implies a semantic weight. Then, the basic idea is that frequency information about a word is probably reflected from the semantic importance of the word. Bilingual co-occurrences, namely alignments in bilingual corpus, are used to catch semantic importance of words (Daille et al. 1994:515-521). However, from the viewpoint of term extraction, ranking methods based on unithood are also intensively studied. For instance, various kinds of statistic information about words co-occurrences which are used to extract promising term candidates that are in the form of collocation (Smadja and McKeown 1990:252-259; Frantzi and Ananiadou (1996:41-46); Shimohata et al. (1997:476-481), are of this type. Among them, C-value (Frantzi and Ananiadou 1996:41-46), entropy (Shimohata et al. 1997:476-481), and Mutual Information (Church and Hanks 1990:22-29) are promising.

Selection subsystem

As for the selection from ranked candidates, we find a very general scheme such as likelihood test (Dunning 1993:62-74). However, we do not find much work that directly treats a term selection process. At the first glance, a selection by the predetermined threshold is, seemingly, simple and powerful. However, the real problem is the way to determine the threshold that works equally well on unseen documents. Since the method using a simple threshold is not the only method, it is a challenging problem to find another promising selection method.

Target of This Paper

In this paper, we report on our experimental results of two automatic term extraction methods. Roughly speaking, “term” means an open compound (Smadja and McKeown 1990:252-259), which is defined as an uninterrupted sequence of words. One extraction method we focus on here is C-value based term extraction (Frantzi and Ananiadou 1996:41-46). The other method we focus on here is based on a certain

kind of statistics about compound word formation (Nakagawa 1997:598-611). Both methods propose the way to rank collocations or compound words according to the importance of each of them. Once all of the term candidates are ranked, then we need a method to select real terms from those ranked candidates. In our experiments, we use a simple threshold selection method and a window method that is introduced later in this paper. Finally we compare and evaluate the results of every combination of these two ranking methods and these two selection methods.

3. Ranking Methods

3.1. *C-Value Based Method*

One of the famous approaches based on statistics about linguistic structure is the ranking method based on C-value (Frantzi and Ananiadou 1996:41-46). They recently updated the definition of C-value and introduced NC-value that is the combination of C-value and the context factor (Frantzi and Ananiadou 1999:145-179). Of course, the new C-value or NC-value might show the better performance. But we adopt the method described by (Frantzi and Ananiaodu 1996:41-46) because the original C-value reflects their original intention. Their term extraction system first extracts all candidates of collocation. Then, it uses the measure they call **C-value** defined by the following formula:

$$\text{C}-\text{value}(a) = \left(\text{length}(a) - 1\right) \times \text{freq}(a) \qquad (1)$$

a is not nested

$$\text{C}-\text{value}(a) = \left(\text{length}(a) - 1\right) \times \left(\text{freq}(a) - \frac{\text{t}(a)}{\text{c}(a)} \right) \qquad (2)$$

otherwise

where a is a collocation, freq(a) is the frequency of occurrence of a in the corpus, t(a) is the number of occurrence of candidates of collocation that contain a, and c(a) is number of the distinct candidates of collocations that contain a. First of all, C-value(a) primarily depends on freq(a) which means how frequently a is used. Thus, if a is a multi-word collocation, C-value shows how stable the collocation a is used. In this sense, C-value(a) indicates unithood of a. But, in fact, things are more complicated. For instance, the collocation “Wall Street” seems to be ranked high in the corpus about finance and business. However, if “Wall Street” almost always appears as a part of “Wall Street Journal” in the corpus, the latter should be ranked higher and the former should be ranked much lower. C-value implements this idea. Precisely speaking, the greater the number of distinct extracted candidate terms that contain a string a, the bigger the C-value of a is. Note that the range of C-value is still confined between the frequency of occurrence of a and zero. Since this characteristic reflects how the writers treat a to some extent, C-value is regarded to indicate termhood as well. Consequently, C-value indicates the combination of unithood

and termhood. Thus, henceforth, we regard C-value based term extraction method as a method indirectly based on both unithood and termhood.

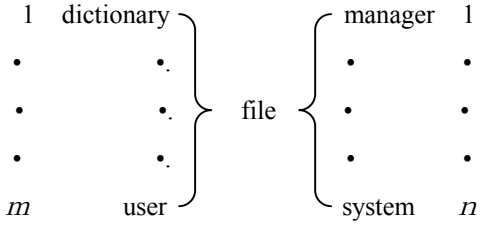
3.2. *Compound Noun based Statistics*

Obviously, the relation between simple terms and complex terms in which they are included is very important. To my knowledge, this relation has not been paid enough attention so far. Nakagawa (1997:598-611) shows a new direction that focuses on the method to use this relation. Here we focus on compound nouns among various types of complex terms. In technical documents, the majority of domain specific terms are complex terms, more precisely compound nouns. In spite of huge number of technical terms being compound nouns, relatively small number of simple nouns contribute to make these compound nouns. Considering this fact, we propose a new scoring method that measures the importance of each simple noun. This scoring method for a simple noun measures how many distinct compound nouns contain the simple noun as their parts in a given document or a set of documents. *Pre* (simple word) and *Post* (simple word) are introduced for this purpose, and defined as follows.

Definition 1

In the given text corpus, *Pre(N)*, where *N* is a noun appearing in the corpus, is the number of distinct nouns that *N* adjoins and make compound nouns with *N*, and *Post(N)* is the number of distinct nouns that adjoin *N* and make compound nouns with *N*.

The key point of this definition is that *Pre(N)* and *Post(N)* do not count the number of total occurrences of words that are adjacent to *N*, but the number of distinct words that adjoin *N* or *N* adjoins. It means that *Pre(N)* and *Post(N)* do not measure surface statistics of compound nouns containing *N*, but do measure how the writer of the technical document interprets *N* and uses it in the document. If a certain word, say *W*, expresses the key concept of the system that the document describes, the writer of the document must use *W* not only many times but also in various ways that include forming and using many compound nouns that contain *W*. This kind of usage really reflects the termhood of that word. In this sense, *Pre* and *Post* very directly measure termhood. Figure 2 shows an example of *Pre* and *Post*.



Pre("file") = m and *Post("file") = n*

Figure 2: An example of *Pre* and *Post*

Next, we extend this scoring method to cover compound nouns. For the given compound noun $N_1N_2....N_k$ where N_is are simple nouns, the scores of importance of $N_1N_2....N_k$, which is called $Imp(N_1N_2....N_k)$, would be defined, for instance, in the following ways.

$$Imp_1(N_1N_2 \ldots N_k) = \prod_{i=1}^{k}((Pre(N_i)+1)\times(Post(N_i)+1)) \quad (3)$$

$$Imp_2(N_1N_2....N_k) = (\prod_{i=1}^{k}((Pre(N_i)+1)(Post(N_i)+1)))^{\frac{1}{2k}} \quad (4)$$

$Imp_1(N)$ directly depends on the length of compound noun N. $Imp_2(N)$ is normalized by the length of N, and does not depend on the length of N.

4. Term Selection Subsystem

We have already explained ranking methods in Figure 1 in the previous section. Then, in the whole system of term extraction depicted by Figure 1, we need to define a selection process, which selects real terms from ranked candidates. As a selection process, we think of two methods: the simple threshold method and the window method.

4.1. *Simple Threshold Method*

It is easy to use a predetermined threshold about the score, like C-value or *Imp*, on ranked candidates to select real terms. Namely, the candidate terms whose C-value or *Imp* score are over that threshold are selected as the real terms and other candidates are abandoned. This selection method is quite simple, but the real difficulty we face in this type of selection is the way to determine the optimum threshold. We do not have a solid theory to determine the threshold which works equally well for various documents at this moment, because each document has distinct characteristics in text length, number of vocabularies, distribution of length of collocation, and so forth. Even in the case where we treat documents of one academic field, we have not yet had any theoretical way to determine the threshold. Then, the only way is to use statistics over a set of documents we are focusing on. As statistics, an average μ and a standard deviation σ of C-value or *Imp* score are essential. Since we have not yet known any thing theoretical about the relation between the threshold *th*, μ, σ and the contents of documents, the easiest way to determine the threshold *th* with μ and σ is given by the following formula:

$$th = a \cdot \mu + b \cdot \sigma \quad (5)$$

where constants a and b are determined to give the best threshold th in terms of recall and precision. Actually, the best a and b depend on individual document. But, if μ and σ express enough amount of information about the given document, we can expect that a and b that are optimized for one document or a set of documents work equally well for other documents. In fact, the best a and b are not heavily different for five Japanese software manuals we use for our experiment.

4.2. *Window Method*

The simple threshold method described in the previous section uses the global statistics like μ and σ but does not use local statistics at all. Then, we focus on the statistical value within the window on ranked candidates as local statistics. In this method, which we call *window method* henceforth, a window with a certain width is moving from the position of the highest ranked term candidate down to the position of the lowest ranked term candidate. For instance, a window of width=3 is depicted in Figure 3.

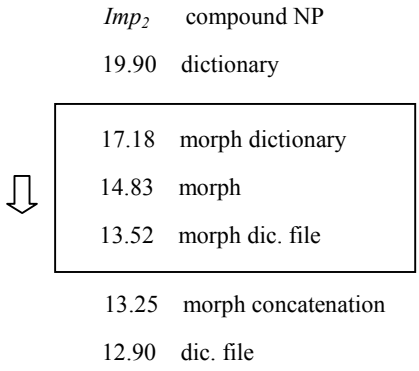


Figure 3: Window with width=3

A position of the window is characterized by the largest value of *Imp* or C-value of the term candidate within the window. For instance, in Figure 3, the window's position corresponds to 17.18. Now we use some statistical values we obtain from the contents of window along with the window moving downwards, to decide whether the nouns in the window is selected as a real term or not. Among several kinds of statistical value, we pay attention to the real term ratio in the window, RTR in short, which is defined as follows.

$$\mathrm{RTR} = \frac{\#(\text{real term in the window})}{\text{window width}} \tag{6}$$

where $\#X$ means the number of members in the set denoted by X.

The reason why we pay attention to RTR is that RTR is, in fact, high in the windows of high *Imp* value. Moreover, the number of real terms increases as the length of document increases. In addition, a number of all of simple nouns and compound nouns in the text also increases as the document becomes longer.

Therefore, RTR is likely to be less dependent on the length of document. We also pay attention to the compound noun ratio in a window, CNR in short, defined as follows.

$$\mathrm{CNR} = \frac{\#(\text{compound noun in the window})}{\text{window width}} \tag{7}$$

The reason why we pay attention to CNR is that the majority of real terms in technical documents are usually compound nouns in the Japanese technical documents we investigated. By considering the nature of RTR and CNR, we reach the following expectation: In the window whose corresponding *Imp* value is high, the majority of simple and compound nouns within the window are real terms, and at the same time, the majority of them are compound nouns, too. Therefore, we expect high relevance between them. In Table 1, we show the correlation coefficients between RTR and CNR for Imp_1 and Imp_2 of five Japanese technical manuals shown in Table 2.

Table 1: Correlation Coefficients between RTR and CNR in a case of a window of width=5

Manual	Coefficient	
	Imp_1	Imp_2
JUMAN	.753	.682
SAX	.628	.591
EGG	.808	.788
HV-F93	.737	.705
Play-Station	.738	.692

Since almost all correlation coefficients between CNR and RTR are higher than 0.6, they are high enough to use CNR value instead of *Imp* values themselves for selection by the given threshold. And from the value of these coefficients, we confirm that among simple and compound nouns having high *Imp* values, the majority of terms are compound nouns. Therefore, what we have to do is to find an optimum, or at least a sub-optimum, threshold of CNR to select the real terms. In the selection process, the term candidate that is located at the center of the window is selected as real term if CNR of the window is larger than the pre-determined threshold; otherwise that candidate is not selected.

5. Experiments

As described previously, we focus here on two ranking methods and two selection methods described in the previous sections, respectively. Then, we made experiments for every combination of ranking method and selection method, namely 1) *Imp* + simple threshold (*Imp*+*Sth*), 2) *Imp* + window method (*Imp*+*Win*), 3) C-value + simple threshold (*Cval*+*Sth*), and 4) C-value + window method (*Cval*+*Win*) . In the rest of this section, we compare the results of these combinations and evaluate these combinations.

Now we explain the details of our experiment. We use five technical manuals written in Japanese shown in Table 2.

Table 2: Manuals written in Japanese used for this research

Manual	Number of sentences	Size (KB)	Number of real terms
JUMAN(software) Morphological analyzer	436	31	106
SAX(software) Parser	433	28	207
EGG(software) Kana-Kanji converter	628	30	108
Home use VCR Mitsubishi HV-F93	1461	69	259
Video Game Machine SONY Play-Station	131	7	39

Terms that are to be extracted, namely real terms are extracted manually in the following way. Three people who use or know well these softwares or hardwares extract manually real terms which, they think, are important to understand and/or characterize the contents of those five manuals. Term Candidates Extraction process shown in Figure 1 is done as follows. Firstly the morphological analyzer JUMAN segments out words from the sentence, and assigns each word a POS tag. Secondly every noun sequence that may contain Japanese particle NO ("of" in English) is extracted as a term candidate. Using both of these term candidates and the real terms above mentioned, we evaluate the previously described combinations, namely *Imp*+*Sth*, *Imp*+*Win*, *Cval*+*Sth*, and *Cval*+*Win*, by recall, precision and F-measure.

As for *Imp* function, we compare Imp_1 and Imp_2, and finally select Imp_2 because it gives the better performance in terms of F-measure:

$$F = \frac{\left(1+\beta^2\right) \times Precision \times Recall}{\beta^2 \times Precision + Recall} \tag{8}$$

where β indicates how much a user is interested in recall as precision. We choose 1.0 as the value of β in our experiment.

As described earlier, our window method has two parameters, which is to say CNR threshold and window width. We tune a CNR threshold and a window width to optimize F-measure. We choose the following four window widths, namely 5, 10, 20 and 30. Then we apply the following 19 CNR thresholds, namely 0.05, 0.1, 0.15, 0.2, and 0.95 for each of those four window widths. Considering the results we get with all the combinations of window width and CNR threshold, we select the combination of the window size and the CNR threshold that gives the best F-measure.

As for the simple threshold method, on the other hand, for the simplicity of threshold selection, we fix a = 1 and tune b in the previously described formula of threshold:

$$th = a \cdot \mu + b \cdot \sigma \tag{9}$$

to minimize F-measure.

In (Frantzi and Ananiadou 1996:41-46), C-value is calculated for word n-grams where n ≥ 2. Here, we decide to use a C-value of uni-gram to rank every n-gram based on C-value in order to compare *Imp* based method with C-value based method. To apply C-value to uni-gram, we change the definition of C-value into the following:

$$\mathrm{C-value}(a) = \mathrm{length}(a) \times \left(\mathrm{freq}(a) - \frac{\mathrm{t}(a)}{\mathrm{c}(a)} \right) \tag{10}$$

Under these experimental conditions, we apply our window method and the simple threshold method to two groups of candidates that are ranked based on *Imp* and C-value, respectively.

We show the results of term extraction of four cases, that is to say *Imp+Win*, *Cval+Win*, *Imp+Sth* and *Cval+Sth*, in Table 3, 4, 5 and 6, respectively. They are the best ones in terms of F-measure. Each table shows the parameters of the selection subsystem such as *b*, window width and the threshold of CNR (Th-CNR), precision (P), recall (R) and F-measure(F) that correspond to the case which gives the best F-measure for each of these five manuals.

Table 3: The results of *Imp+Win*

Manual	Window width	Th-CNR	R	P	F
JUMAN	20	0.6	0.491	0.658	0.562
SAX	30	0.1	0.507	0.507	0507
EGG	30	0.6	0.472	0.405	0.436
HV-F93	5	0.3	0.602	0.495	0.544
Play-Station	20	0.4	0.615	0.5	0.552
Average			0.537	0.513	0.520

Table 4: The results of *Imp+Sth*

Manual	*b*	R	P	F
JUMAN	-0.3	0.519	0.509	0.514
SAX	-0.75	0.541	0.5	0520
EGG	-0.2	0.556	0.345	0.427
HV-F93	-0.7	0.629	0.452	0.526
Play-Station	-0.95	0.615	0.5	0.552
Average		0.572	0.461	0.508

Table 5: The results of *Cval+Win*

Manual	Window width	Th-CNR	R	P	F
JUMAN	5	0.35	0.319	0.708	0.44
SAX	20	0.2	0.691	0.464	0.555
EGG	5	0.35	0.741	0.273	0.399
HV-F93	20	0.4	0.741	0.339	0.465

Play-Station	10	0.3	0.667	0.413	0.509
Average			0.631	0.439	0.474

Table 6: The results of *Cval+Sth*

Manual	*b*	R	P	F
JUMAN	0.1	0.425	0.584	0.492
SAX	-0.6	0.696	0.45	0.546
EGG	-0.2	0.556	0.345	0.427
HV-F93	-0.7	0.629	0.452	0.526
Play-Station	-0.95	0.615	0.5	0.552
Average		0.572	0.461	0.508

As indicated in Tables 3, 4, 5 and 6, *Imp+Win* shows the best F-measure. Moreover, *Imp* based methods outperform C-value based methods, whichever selection subsystem is employed.

In actual applications, we have to deal with unseen documents. That means that we could not use the optimized parameters described in Tables 3, 4, 5 and 6. To estimate the performance of proposed systems for unseen documents, we use the average values of the parameters, and show the results in Tables 7, 8, 9 and 10. The general tendency is almost the same as the best F-measure cases shown in Tables 3, 4, 5 and 6 where P, R, and F stand for Precision, Recall and F-measure, respectively. Precisely speaking, *Imp* based methods outperform C-value based methods. The degradations of F-measure are less than 5% in every case except for *Cval+Sth* whose degradation is 9.6%. This means that all of these combinations are expected to work well for unseen documents, at least, for technical manuals.

Table 7: The results of *Imp+Win*

Window width=22, CNR=0.376

Manual	R	P	F
JUMAN	0.443	0.580	0.503
SAX	0.372	0.583	0.454
EGG	0.481	0.374	0.421
HV-F93	0.521	0.491	0.506
Play-Station	0.487	0.559	0.521
Average	0.460	0.512	0.481

Table 8: The results of *Imp+Sth*

b=-0.58

Manual	R	P	F
JUMAN	0.639	0.280	0.390
SAX	0.598	0.468	0.525
EGG	0.585	0.383	0.463
HV-F93	0.564	0.524	0.543
Play-Station	0.488	0.526	0.506
Average	0.575	0.436	0.485

Table 9: The results of *Cval+Win*

Window width=12, CNR=0.32

Manual	R	P	F
JUMAN	0.769	0.251	0.378
SAX	0.726	0.334	0.457
EGG	0.726	0.291	0.415
HV-F93	0.718	0.424	0.533
Play-Station	0.594	0.475	0.528
Average	0.707	0.355	0.462

Table 10: The results of *Cval+Sth*

b=-0.35

Manual	R	P	F
JUMAN	0.670	0.311	0.425
SAX	0.478	0.518	0.497
EGG	0.722	0.263	0.385
HV-F93	0.483	0.398	0.436
Play-Station	0.359	0.378	0.368
Average	0.542	0.374	0.422

For more precise comparison among four combinations, we show recall-precision relations of *Imp+Win*, *Imp+Sth*, *Cval+Win* and *Cval+Sth* for each of these five manuals in Figures 4, 5, 6, 7 and 8, respectively.

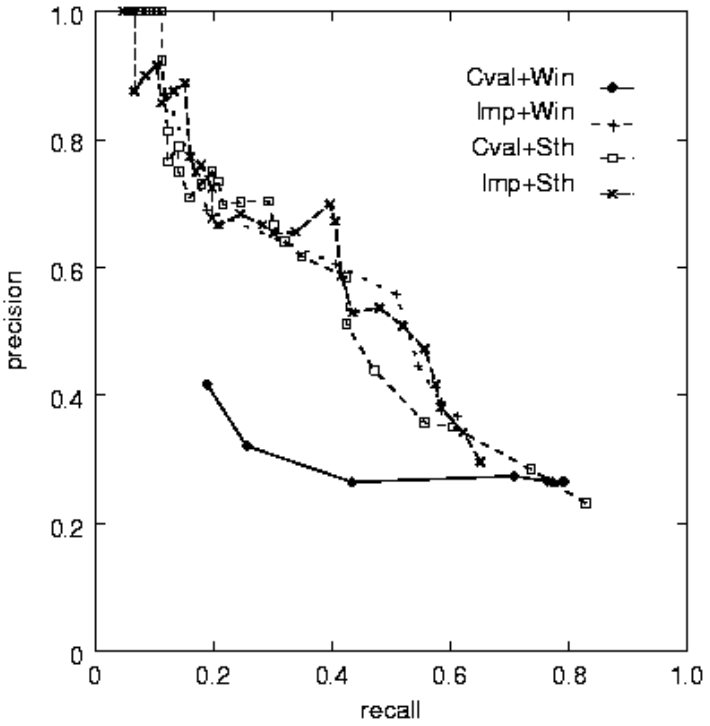


Figure 4: Recall-Precisions for JUMAN

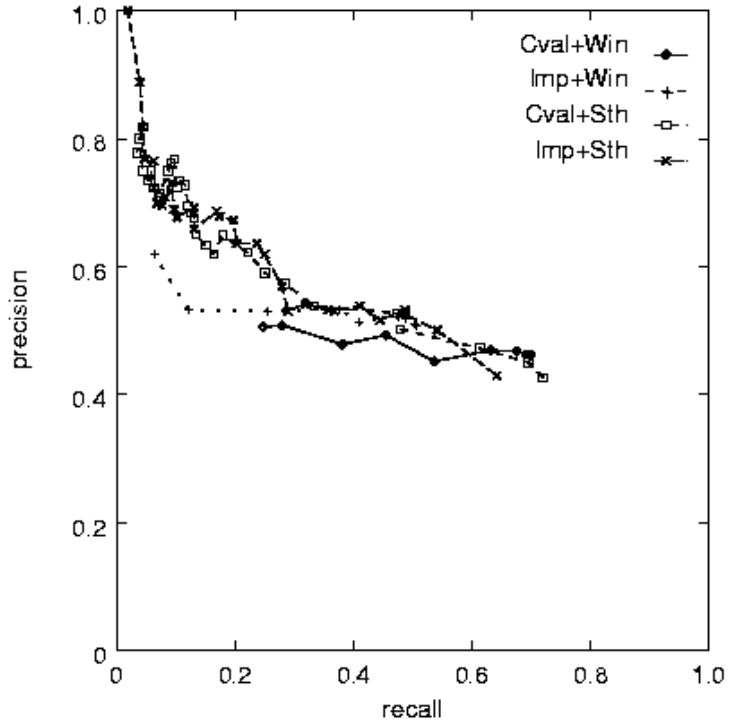


Figure 5: Recall-Precisions for SAX

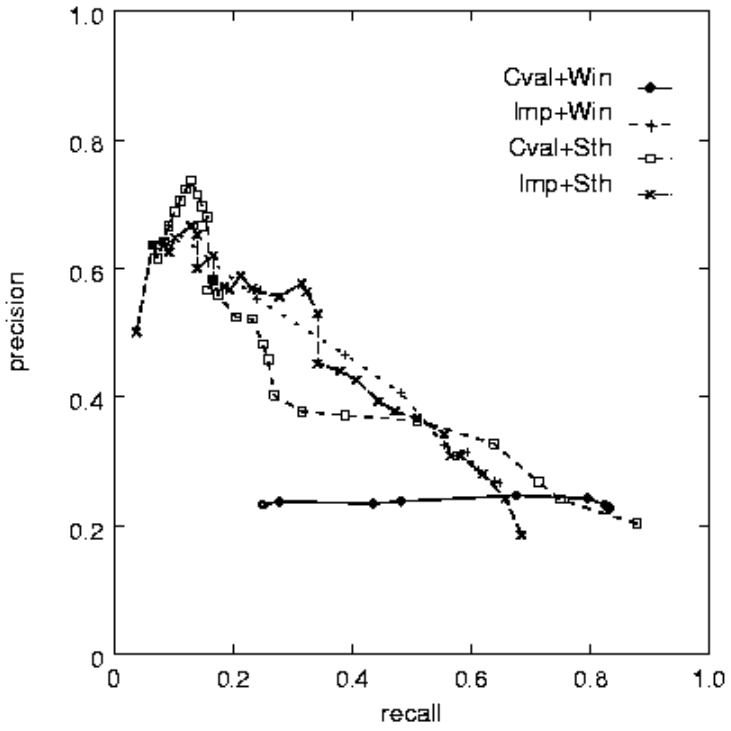


Figure 6: Recall-Precisions for EGG

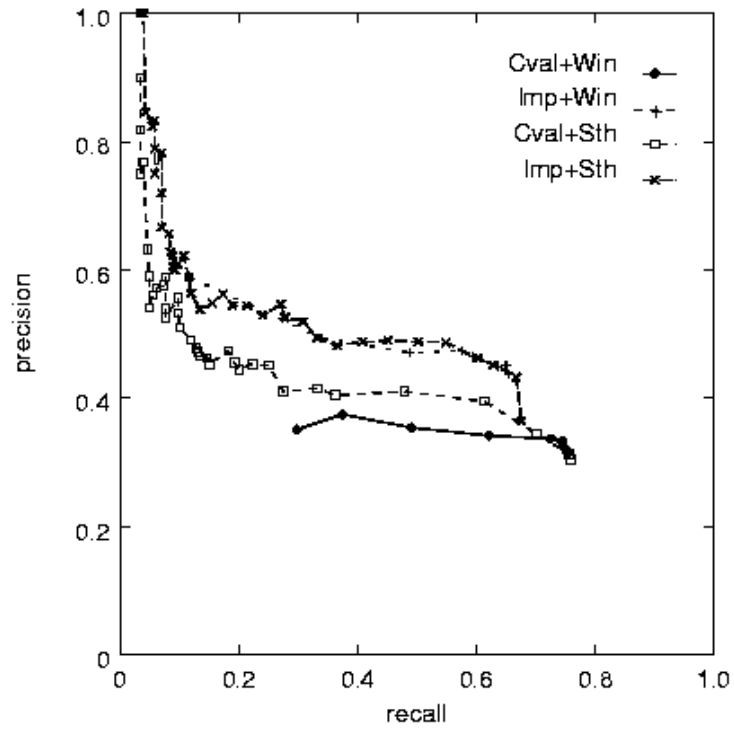


Figure 7: Recall-Precisions for HV-F93

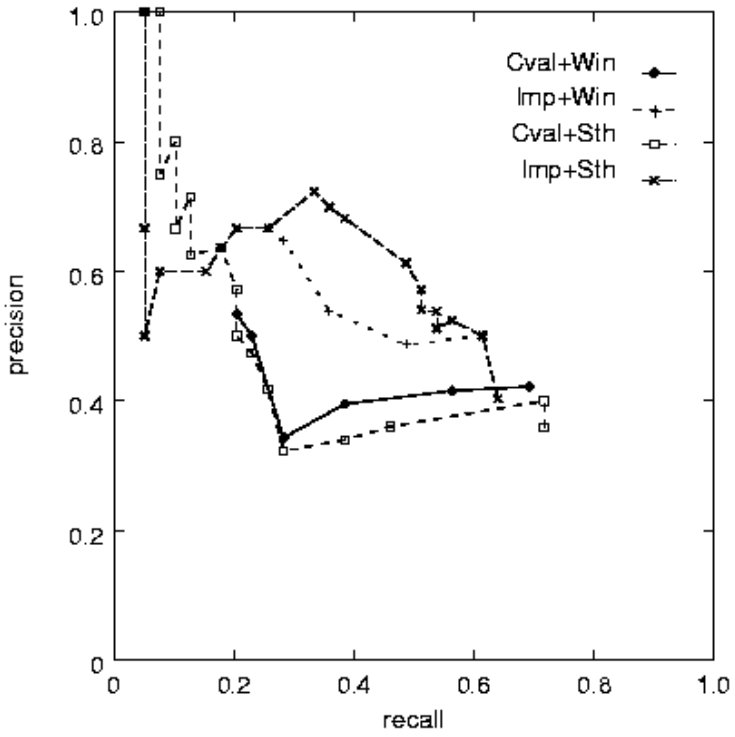


Figure 8: Recall - Precision for Play-Station

In these figures, the window width is 30 and the threshold of CNR, which corresponds to Th-CNR in Tables 3 through 10, varies from 0.1 to 0.9 in the window method, and the parameter *b* of simple threshold method varies from -3 to +3. As you know from these figures, *Imp* based methods are superior to C-value based methods. However, the difference between *Imp* based methods and *Cval+Sth* is not significant in JUMAN, SAX, and EGG. *Cval+Sth* is far worse than *Imp* based methods in HV-F93 manual and Play-Station manual. *Cval+Win* is far worse than other three methods in all manuals. We will describe the reason for these phenomena later on. In brief, *Imp* based ranking method that is directly based on termhood slightly outperforms C-value based ranking method that is indirectly based on both of unithood and termhood. It is needless to say that these experimental results could not be generalized. The best ranking method could depend on many factors including language, academic area, size of corpus, etc.

Next, we are going to focus on extracted terms themselves for each ranking method. As an example, we show the terms extracted from the manual of JUMAN (Japanese morphological analyzer software). We show the terms extracted by both of Imp_2 + Window Method and C-value + Window Method, the terms extracted exclusively by Imp_2 + Window Method, and the extract terms exclusively by C-value + Window Method in the following. Since the document itself is written in Japanese, the extracted terms are also Japanese. For the convenience of nonnative readers, we show the English translations of these, too.

Parts of terms extracted from a Japanese manual by both of Imp_2 based and C-value based ranking methods

C Ban (C version) / JUMAN sisutemu (JUMAN system) / Prolog Ban (Prolog version) / Gurahu Kouzou (graph structure) / Kosuto (cost) / Kosuto Keisan (cost calculation) / Kosuto Haba (cost band width) / Sisutemu Zisho (system dictionary) / Sisutemu Hyoujun Zisho (system standard dictionary) / Sisutemu Hyoujun BUnpou (system standard grammar) / Yuuza Zisho (user dictionary) / Imi Zisho (semantic dictionary) / Kakutyousi (extension) / Katuyou (inflection) / Katuyou Kankei Zisho (inflection relation dictionary) / Kstuyou-kei Mei /(inflection name) Katuyou Zisho (inflection dictionary) / Keitaiso (morphology) / Keitaiso Kosuto (morphology cost) / Keitaiso Kaiseki (morphological analysis) / keitaiso Kaiseki Puroguramu (morphological analysis program) / Keitaiso Kouzou (morphological structure) / keitaiso Zisho (morphology dictionary) / Keitaiso Zisho Fairu (morphology dictionary file) / Keitaiso Jouhou (morphology information) / Keitai Hinshi (morph part of speech) / Keitai Hinsi Bunrui Zisho (morph part of speech classification dictionary) / Keitai Hinshi Mei (morph part of speech name) / Midasigo (entry word) / Go (word) / .....

• Total number of extracted terms is 53.

Terms exclusively extracted by Imp_2 based ranking method

.jumanrc Fairu (.jumanrc file) / Entori (entry) / Opushon Teigi (option definition) / Gurahu (graph) / Hasshu Teeburu (hashing table) / Katuyoukei (inflection form) / Kihonkei (root form) / Gobi (suffix) / Hyousou (surface) / Henkan (transformation)/

• Total number of extracted terms is 10.

Terms exclusively extracted by C-value ranking method

Opushon Teigi Fairu (optional definition file) / Keitaiso Bunpou (morphology grammar) / Kousetu Jouhou (postfix information) / Kouzou (structure)/ Soku-Jou (lattice like)/ Takubo Bunpou (Takubo grammar)/ Rensetu Kanousei (connection possibility)/

•Total number of extracted terms is 7.

At the first glance the majority of terms, 75.7%(=53/70), are extracted by both of C-value based ranking method and *Imp* based ranking method. This means that these two ranking methods based on different concepts, say directly based on termhood and indirectly based on both termhood and unithood, actually give very similar results. It is too early to say, but unithood and termhood has strong correlation in terms of ranking candidates of terms, which are collocations or compound nouns. The theoretical background of this correlation is, at this moment, an open problem.

Focusing on the actual mechanism of these two methods, it is much more important to investigate the terms exclusively extracted by each ranking method. Six out of seven terms extracted exclusively by C-value based ranking are collocations, in other words, compound nouns in this case. It is a reasonable result because C-value is originally developed to rank not simple nouns but collocations. We forced to change its original definition in order to score simple nouns. We once again write our new definition of C-value here:

$$\mathrm{C-value}(a) = \mathrm{length}(a) \times \left(\mathrm{freq}(a) - \frac{\mathrm{t}(a)}{\mathrm{c}(a)} \right) \tag{11}$$

From this formula, it is known that a simple noun, which is a part of many compound nouns, gets a high score of $\left(\mathrm{freq}(a) - \frac{\mathrm{t}(a)}{\mathrm{c}(a)} \right)$ of (11). However, since its length is 1, it does not have a high score when compared to longer compound nouns. A simple noun that is not a part of many compound nouns obviously does not have a high score by the definition of C-value. This is the reason why C-value based ranking does favor longer collocations, in other words it does favor compound nouns. On the contrary, seven out of ten terms extracted by Imp_2 based ranking are simple nouns. *Imp* is calculated with *Pre(N)* and *Post(N)*, which express how important the simple noun *N* is. Especially for compound nouns, Imp_2 does not depend on the length of a compound noun by its definition. Therefore, simple nouns are treated as equally well as longer compound nouns. This is the reason why Imp_2 based ranking method favors a simple noun more than C-value based ranking method does. In brief, whether simple nouns are preferred to compound nouns or not does not depends on the dichotomy of unithood and termhood, but on whether a scoring method treats simple nouns and compound nouns equally or not. In this sense, it is said that *Imp* based ranking method has high flexibility because it has many variations for the definition of *Imp* that is defined with *Pre* and *Post*. To conclude this section, we answer the pending questions, namely 1) why *Cval+Win* is the worst, and 2) why *Cval+Sth* is as equally bad as *Cval+Win* especially for HV-F93 and Play-Station manuals. We answer the first question at first. Since C-value is low for simple nouns in general, there remain quite a few of simple nouns that are to be selected in low C-value area. Moreover, in that low C-value area, there remain very few compound nouns. Then, in that area, if we put high threshold of CNR, we fail to select many terms that are simple nouns. On the contrary, if we put low threshold of CNR, we end up with picking up many non-real terms, because the majority of candidates in that low C-value area are not real terms. In short, the algorithm of the window method does not work well in low C-value area. Next we answer the second question. The users of these two equipments, HV-F93 and Play-Station, are not engineers but ordinary people. Consequently, many of important terms are simple nouns. Thus, C-value based method may fail to give high score to the real terms that are simple nouns.

6. Conclusions

We have first explained a dichotomy of unithood and termhood. We explain C-value based method which is a ranking method indirectly based on both of termhood and unithood. Then, we explained the ranking method that uses statistics of compound noun structure, called *Imp* that is directly based on termhood. We also explain the simple threshold method and the window method that are used to select real terms among ranked term candidates. We experimentally estimate *Imp* based method and C-value based method for Japanese technical manuals. Both are showing the almost same result in precision, recall and F-measure. But the sets of terms extracted by two methods are little bit different. In this sense, how directly an extraction method is based on termhood is not only characteristic of term extraction, and still there remain many linguistic features from the viewpoint of term extraction.

We are now conducting the experiment of term extraction from English documents. A term extraction process for English documents is basically the same as the Japanese case described above. The difference is in the term candidate extraction subsystem. In English document cases, at first we apply the input document a part of speech tagger such as (Brill 1994b) to assign a part of speech tag to each morpheme. In Japanese case, a compound noun is a consecutive sequence of nouns, which may include particle NO ("of" in English) between nouns. However, terms in English often take a pattern of adjective + noun, noun + preposition + noun, etc as well as a noun sequence. So, we need a linguistic filter to pick up those patterns exclusively. To apply a stop-list is also necessary to exclude words that are not suitable components of terms of the target domain. The easiest way, which we actually try to use, to apply *Imp* function to texts is to pick up sequences of words that are not interrupted by any word in the stop-list. We have already applied *Imp* function based term extraction method to uninterrupted words sequences to small English corpus. The result seems to be not bad, but to evaluate our method by processing much larger corpus is our future problem. Seeking better definition of *Imp* function experimentally and comparison of other different term extraction methods using larger scale corpora are also our future problems.

References

Ananiadou, S. 1994. "A methodology for automatic term recognition". In *Proceedings of 15th International Conference on Computational Linguistics, 1034*-1038.

Brill, E. 1994a. "Some advances in transformation-based part-of-speech tagging". In *Proceedings of 11th National Conference on Artificial Intelligence*, 722-727.

Brill, E. 1994b. "Supervised part of speech tagger". *http://www.cs.jhu.edu/~brill/*.

Church, K.W. and Hanks, P. 1990. “Word association norms, mutual information, and lexicography” *Computational Linguistics*, 16(1):22-29.

Daille, B., G.aussier, E. and Lange, J.M. 1994. “Towards automatic extraction of monolingual and bilingual terminology”. In *Proceedings of 15th International Conference on Computational Linguistics*, 515-521.

Dunning, T. 1993. “ Accurate methods for the statistics of surprise and coincidence”. *Computational Linguistics*, 19(1):62-74.

Frantzi, T.K. and Ananiadou, S. 1996. “Extracting nested collocations”. In *16th Proceedings of 15th International Conference on Computational Linguistics*, 41-46.

Frantzi, T.K. and Ananiadou, S. 1999. “The c-value/nc-value method for atr”. *Journal of Natural Language Processing*, 6(3): 145-179.

Fujii, H. and Croft, W.B. 1993. “A comparison of indexing techniques for Japanese text retrieval”. In *Proceedings of 16th International Conference on Research and Development in Information Retrieval*, 237-246.

Hisamitsu, T. and Nitta, Y. 1996. “Analysis of Japanese compound nouns by direct text scanning”. In *16th Proceedings of 15th International Conference on Computational Linguistics*, 550-555.

Kageura, K. and Umino, B. 1996. “Methods of automatic term recognition: a review”. *Terminology*, 3(2):259-289.

Lam, W., Wong, C.Y. and Wong, K.F. 1997. “Performance evaluation of character, word and n-gram-based indexing for Chinese text retrieval”. In *Proceedings of the Second International Workshop on Information Retrieval With Asian Languages*, 68-80.

Matsumoto, Y., Kurohashi, S., Yamaji, O., Taeki, H. and Nagao, M. 1996. *Instruction Manual of Japanese Morphological Analyzer JUMAN3.1*. Nagao Lab. at Kyoto University.

Nakagawa, H. 1997. “Extraction of index words from manuals”. In *Conference Proceedings of Computer-Assisted Information Searching on Internet*, 598-611.

Shimohata, S., Sugio, T. and Nagata, J. 1997. “Retrieving collocations by co-occurrences and word order constraints”. In *Proceedings of 35th Annual Meetings of the Association for Computational Linguistics*, 476-481.

Smadja, F.A. and Mckeown, K.R. 1990. “Automatically extracting and representing collocations for language generation”. In *Proceedings of the 28th Annual Meetings of the Association for Computational Linguistics*, 252-259.

Zhai, C. and Evans, D.A. 1996. “Noun-phrase analysis in unrestricted text for information retrieval”. In *Proceedings of 34th Annual Meetings of the Association for Computational Linguistics*, 17-23.