Adaptation of a Keyphrase Extractor for Japanese Text

Jérôme Mathieu
Institute for Information Technology, National Research Council of Canada
Ottawa, Ontario, Canada
jerome.mathieu@iit.nrc.ca

Abstract

This paper presents some statistical observations relevant to Japanese keyphrase extraction, as well as the details of the implementation of a keyphrase extraction algorithm (called Extractor) for Japanese documents. Parts of the algorithm include an efficient method of extracting the keyphrase candidates, a way to pinpoint the most probable keyphrases using contextual information, a technique to find the main ideas conveyed in the text, and a way to express those ideas using extracted phrases. Finally, a comparison with the English and French versions of Extractor will be presented.

1. Introduction

One of the research areas of the Interactive Information Group at the Institute for Information Technology of the National Research Council Canada is algorithms and software for text analysis and retrieval. The current research projects include Extractor[1], a new software tool that extracts keyphrases from a document (Turney, 1997). The demand for this kind of technology is increasing, as the quantity of digital information that is available is becoming overwhelming. To manage the profusion of information, we need tools to automatically summarise text documents.

A tool that can automatically extract keyphrases from text can enable many different types of information retrieval and analysis. Potential applications for a keyphrase extractor include metadata creation, content highlighting, indexing, interactive query refinement, web log analysis, and a "what's related" function for search engines.

We apply techniques from machine learning to the problem of automatically extracting keyphrases from text. We approach the problem as a supervised learning task. Another paper (Turney, 1999) describes our algorithm for keyphrase extraction with English text. In this paper, we focus on the implementation of an efficient and effective way to extract keyphrases from Japanese text.

This paper begins with a brief introduction to Extractor. Then it explains the main characteristics of Japanese text in contrast to English text. It presents some statistics about some sample Japanese documents that were used for the training of the software. It also presents the details of the keyphrase extraction process for Japanese text, which includes the parsing of the text, the fragmentation of the candidate phrases, the scoring mechanism, and the filtering of the candidate phrases. The paper then compares the Japanese version of Extractor with the English and French versions. Finally, some words about our future work and a conclusion are presented.

2. Extractor

Extractor is software that extracts keyphrases from a given document. The extracted keyphrases are intended to be similar to the keywords provided by the author of an article for a journal.

Before Extractor can be used on its own, it has to be trained. Extractor learns to imitate the way the authors write keyphrases, by using sample documents with keyphrases written by the original authors. Extractor is guided during the training process by a genetic algorithm to improve its performance. The guidance consists of the tuning of Extractor's parameters. The number and the kind of parameters change depending on the language. In this case, for Japanese text, Extractor has 24 parameters (Table 1).

[1] Official Mark of the Government of Canada. A demonstration version is available from "http://extractor.iit.nrc.ca".

Table 1: List of the parameters used for the extraction of keyphrases from Japanese documents.

Parameter name	min	max	sample value	Description
desired_number_phrases	4	19	7	The number of desired phrases
PJ_kanji_frag_length	1	11	2	Length of kanji fragment
PJ_kata_frag_length	2	23	4	Length of a katakana fragment
PJ_roman_frag_length	2	100	1024	Length of a roman fragment
PJ_num_working	30	157	60	Length of working list
PJ_first_low_thresh	1	1000	40	Definition of "early" occurrence (*)
PJ_first_high_thresh	1	4000	400	Definition of "late" occurrence (*)
PJ_first_low_factor	1	15	2	Factor applied to the "early" occurrence
PJ_first_high_factor	0.01	1	0.65	Factor applied to the "late" occurrence
PJ_weight_kanji	0.01	1	1	Weight of kanji key phrases
PJ_weight_kata	0.01	1	1	Weight of katakana key phrases
PJ_weight_kankat	0.01	1	1	Weight of kankat key phrases
PJ_weight_roman	0.01	1	4	Weight of roman key phrases
PJ_kanji_length_peak	0.001	1	0.287	Height of the peak for kanji length distribution
PJ_kanji_length_best	2	5	4	The best length for kanji key phrases
PJ_kanji_length_slope	0.001	1	0.041	The slope for the kanji length distribution
PJ_kata_length_peak	0.001	1	0.102	Height of the peak for kata length distribution
PJ_kata_length_best	4	11	7	The best length for kata key phrases
PJ_kata_length_slope	0.001	1	0.0136	The slope for the kata length distribution
PJ_kankat_length_peak	0.001	1	0.09	Height of the peak for kankat length distribution
PJ_kankat_length_best	5	12	8	The best length for kankat key phrases
PJ_kankat_length_slope	0.001	1	0.0075	The slope for the kankat length distribution
PJ_roman_length_best	5	12	8	The best length for the non-abbreviated roman key phrases
PJ_roman_length_slope	0.001	1	0.1	The slope for the non-abbreviated roman length distribution

(*): Measured in number of raw-candidate phrases.

3. Characteristics of Japanese text

This section is a brief introduction to Japanese text.

3.1. The character sets

Apart from the Arabic numbers, the punctuation signs, and other special characters, there are four main character classes used widely in Japanese text. They are kanji, hiragana, katakana and roman characters (Table 2).

Table 2: Examples of characters.

Character class	Example
Kanji	漢字
Hiragana	ひらがな
Katakana	カタカナ
Roman	Roman

Kanji characters are ideograms. They originally came from China, but a few were also created in Japan. Each kanji has by itself a meaning. In a text, they can be used alone, or in combination with other kanji to form words. The way to read kanji characters is complicated, because, depending on the context, the same kanji can be pronounced in different ways. There are about 2000-3000 kanji characters that are used in daily life.

Hiragana and katakana characters belong only to the Japanese language. They are both sets of phonetic characters.

The main role of hiragana characters is to join words together to form a sentence. They can be considered as the base of the Japanese grammar. A special use of hiragana can be observed in children's books, where no kanji are used, but only hiragana, since it is easier for beginners to read.

As for katakana characters, they are mainly used for writing the pronunciation of foreign words, or Japanese words that have a foreign origin.

Finally, roman characters are sometime found in Japanese text. They are usually used for writing English acronyms, abbreviations, and proper names.

3.2. The Japanese Sentence

Japanese sentences are very different from English ones. Among the differences, there are no spaces that separate the words, and sentences have a subject-object-verb structure instead of a subject-verb-object structure. Also, Japanese sentences use hiragana particles, which helps to identify the part of speech of the preceding phrase within the sentence.

4. Statistical information

We gathered from the web 200 Japanese documents. Each had the original author's keyphrases accompanying it. Among those documents, 150 were journal articles, and 50 were reports written by teachers. We carefully analysed those sample documents in order to get a better understanding of how the computer could guess which phrases from the text are keyphrases that an author would be inclined to write. Part of the analysis revealed the following statistics.

4.1. The authors' keyphrases that appear in the documents

The first thing we wanted to know is if the authors write keyphrases that actually appear in the text. We observed that 81.5% of the authors' keyphrases appear in the corresponding document. From this, we see that if we carefully choose the phrases from the document, we could match most of the authors' keyphrases.

Even though it is theoretically possible to extract from the text only the phrases that match the authors' keyphrases, it is not possible, in practice. Even a well-informed reader would not be able to always choose only and exactly the same keyphrases as the original author.

4.2. Categorising the authors' keyphrases

In order to have an objective perception of the actual composition of a typical keyphrase, we looked at the classes of characters used in the authors' keyphrases. More precisely, we categorised each keyphrase depending on the character class used. We then looked at the distribution of each category of keyphrase (Table 3).

Table 3: This table shows the most important keyphrase categories and their corresponding proportions.

Keyphrase Category	Character classes					%	
	Kanji	Katakana	Roman*	Hiragana	Special		
(i) Kanji-only	o					42	89%
(ii) Katakana-only		o				17	
(iii) Kanji-katakana	o	o				15	
(iv) Roman-only			o			15	
(v) Kanji-hiragana	o			o		6	
(vi) Kanji-roman	o		o			1	
(vii) Kanji-kata-hira	o	o		o		1	
(viii) Katakana-rom		o	o			1	
(all others)	?	?	?	?	?	6	
Total (without approximations):						100	

* The roman character class includes the space character.

Our analysis revealed that 42% of the keyphrases provided by the authors are expressed using only kanji characters. We also saw a large majority (89%) of the authors' keyphrases is formed with (i) kanji-only, (ii) katakana-only, (iii) kanji and katakana, or (iv) roman-only characters. In the following sections, you will understand why we chose to extract only those categories.

4.3. The preceding and following characters of an author's keyphrase

It is easy to differentiate the words in English, because of the white spaces between them. However, in Japanese text, there are no white spaces to differentiate the words. Therefore we decided to look closely at the characters that immediately precede and follow the authors' keyphrases, to isolate the possible keyphrases.

In order to have a better overview of the trend, we grouped the characters in four mutually exclusive character classes: (a) special, (b) hiragana, (c) distinguishable, and (d) non-distinguishable.

The special class includes characters like new-line, tab, punctuation signs and numbers. The hiragana class includes all hiragana characters. The distinguishable and non-distinguishable classes vary depending on the keyphrase category. If the keyphrase category is roman-only, then the distinguishable characters are kanji and katakana, and the non-distinguishable characters are roman characters. If the keyphrase category is kanji-only, katakana-only, or kanji-katakana, then the distinguishable characters are roman characters, and the non-distinguishable characters are katakana and kanji characters.

Figure 1: This graph shows, for each keyphrase category, the distribution of *the characters that immediately precede* an author keyphrase.

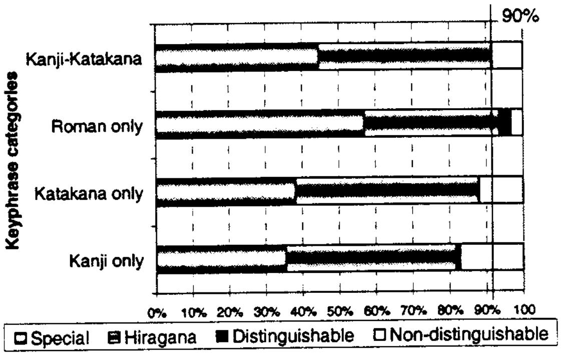
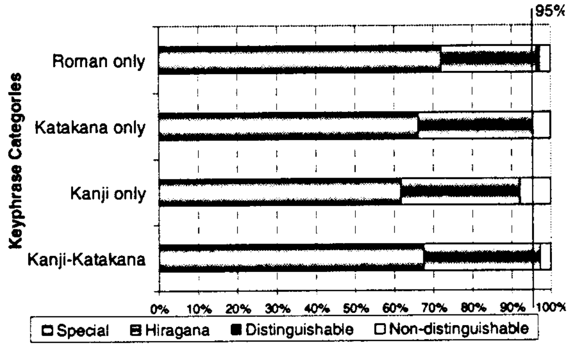


Figure 2: This graph shows, for each keyphrase category, the distribution of the *characters that immediately follow* an author keyphrase.



From Figure 1 and Figure 2, you can see that about 93% of the authors' keyphrases have as preceding or following character, either a special character, a hiragana character, or a distinguishable character.

4.4. The length of the authors' keyphrases

After some preliminary tests, we realised that the length of the keyphrase, when using our parsing method, needs to be considered for properly choosing the best keyphrase (the method will be described in the next section). So, in order to capture the distribution pattern of the length of the keyphrases, we looked at the length of the authors' keyphrases (Figure 3, 4, 5, and 6). We observed a clear difference between the distribution patterns of the length of the keyphrases depending on the category.

If we look at the length of the kanji keyphrases provided by the authors (Figure 3), we observe that the best length for this kind of keyphrase is 4. We also see that the length usually varies between 2 and 7.

By looking at the length of the katakana keyphrases provided by the authors (Figure 4), we observe that the best length for this kind of keyphrase is 7. Also, we see that the length usually varies between 3 and 13.

By looking at the length of the roman keyphrases (Figure 5), we see two peaks at length 3 and 4, and then, we see another peak at 8 and the length varies between 3 and 17. The two peaks can be explained by the extensive use of roman acronyms as keyphrases for Japanese text.

By looking at the length of the author keyphrases that contain katakana and kanji characters (Figure 6), we see that the length usually varies between 5 and 14.

Figure 3: Distribution of the *length* of the *kanji* keyphrases provided by the authors.

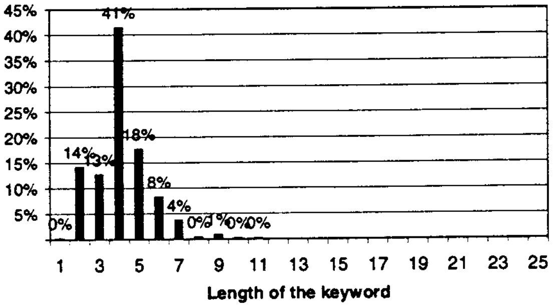


Figure 4: Distribution of the *length* of the *katakana* keyphrases provided by the authors.

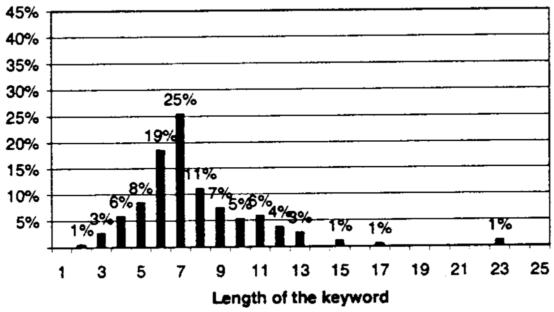


Figure 5: Distribution of the *length* of the *roman* keyphrases provided by the authors.

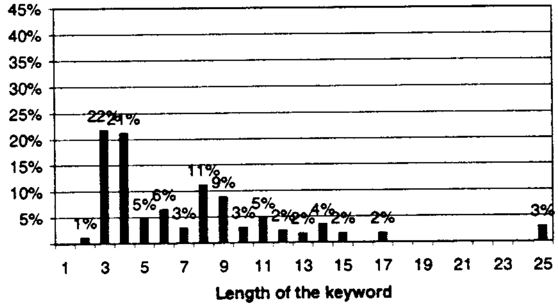
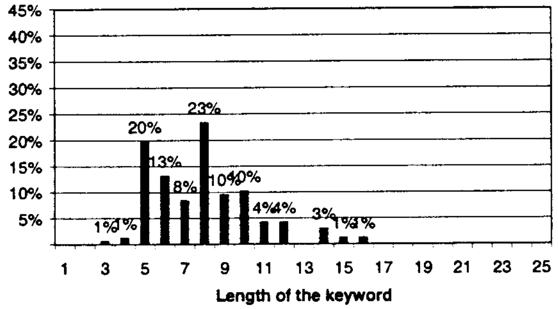


Figure 6: Distribution of the *length* of the *katakana_+ kanji* keyphrases provided by the authors.



5. Keyphrase extraction process

This section presents the details of our keyphrase extraction process for Japanese text.

5.1. Step 1: Parsing

In the parsing stage, we extract all the candidate keyphrases from the text, and gather contextual information concerning them.

In order to extract the candidate keyphrases, we could have used other methods than the one that is presented in this paper, but for the sake of performance, we created a parsing method adapted to keyphrase extraction, instead of using one meant to be used for document indexing (Ogawa, Bessho and Hirose, 1993; Ogawa and Matsuda, 1997).

5.1.1. Dividing the text into *raw-candidates*

The first task of the parsing is to divide the text into *raw-candidates*. By the term *raw-candidate*, we mean a string of characters that contains at most only one candidate keyphrase.

Table 4: *Raw-candidates* separation rules (not exhaustive).

Never start a *raw-candidate* with	
	Examples
White spaces, hiragana, katakana middle dot, katakana/hiragana prolonged sound mark, closing parenthesis, unusual characters, etc.	<white> に ・ ー)」 ,.;: …

Separate when reading a change		
From	*To*	*Examples*
Hiragana	Kanji, katakana, roman, white-space, number	…に|行… …に|ア… …が|Photo… …が| Photo… …に|２時間…
Roman	Kanji, katakana	…CDROM|版… …ftp|サイト…
Kanji, Katakana	Roman, white space	…研究所|ftp… …研究所| ftp… …プログラム|ATOK… …プログラム| ATOK…

Separate when reading	
	Examples
Closing parenthesis	)
Closing Japanese quote	」
Punctuation, Unusual characters	;:,.\“”#$*& etc.

The division is made by applying about two dozen rules, which the computer follows to separate the text into *raw-candidates* (Table 4). Those rules came from the idea of dividing the text when a shift from one character class to another is encountered. (See Figures Figure 1 and Figure 2.) Those rules were refined by trial and error.

Due to our separation rules, the *raw-candidates* are believed to be of the format given in Table 5.

Table 5: Format of a raw-candidate.

Raw-candidate format		
"[<#> <N>] [Time] [P] [Candidate] [S] [Hira]"		
tag	*Description*	*Example*
#	Numerals	'1', '一' (One), '2', '二' (Two), etc.
N	Numeratives	'バイト' (Byte), 'ビット' (Bit), etc.
Time	Time-related stop phrase	'来週' (Next week).
P	Prefix	'各' (Each).
Candidate	Candidate keyphrase	'日本' (Japan)
S	Suffix	'上' (Above).
Hira	Hiragana characters.	'である' (Is).
'[' and ']', indicate that this part of the raw-candidate format may not necessarily appear in the actual raw-candidates. '<' and '>', indicate that this part of the raw-candidate format has to appear.		

5.1.2. Extracting the candidates from the *raw-candidates*

From the *raw-candidate*, we select the candidate keyphrase by disregarding the numerals, the numeratives, the time-related stop phrases, the prefixes, the suffixes and the trailing hiragana characters.

Then, if we select a candidate keyphrase from a *raw-candidate* phrase, we keep its canonical form (i.e. full-width and lowercase characters) and keep also the candidate keyphrase if the canonical form is not exactly the same.

We also dispose of all the candidate phrases that we consider too short to be keyphrases (i.e. if length is less than two characters for kanji keyphrases and if length is less than three characters for roman keyphrases).

Before throwing away the *raw-candidates*, we inspect each *raw-candidate* that contained a candidate keyphrase to see if we can get some useful information about the candidate.

5.1.3. The "sahen-verbs"

One of the types of contextual information that the algorithm takes note of involves the verbal nouns. In Japanese, we can guess whether a katakana-only or kanji-only phrase is a verb or not, by looking at the trailing hiragana characters. If the hiragana characters are 'する' (suru), or one of its conjugated forms, there is a high probability that the phrase is used as a verb. Extractor keeps track of which phrases are likely to be verbs and will not display them at the end of the extraction process.

5.1.4. The propitious contexts for keyphrase

After some informal analysis of the authors' keyphrases in the documents, we believe that each of the following patterns indicates a propitious context for a good keyphrase:

- A candidate keyphrase followed by 'の' (no);
- a candidate keyphrase followed by 'を' (o);
- a candidate keyphrase followed by 'に' (ni);
- a candidate keyphrase followed by 'と' (to);
- a candidate keyphrase followed by 'が' (ga);
- a candidate keyphrase between two Japanese quotation marks: '「' and '」';
- a candidate keyphrase surrounded by two parentheses: '(' and ')';
- a candidate keyphrase surrounded by white spaces like spaces, new-line, tab, etc.;
- a candidate keyphrase, which is written only in roman characters.

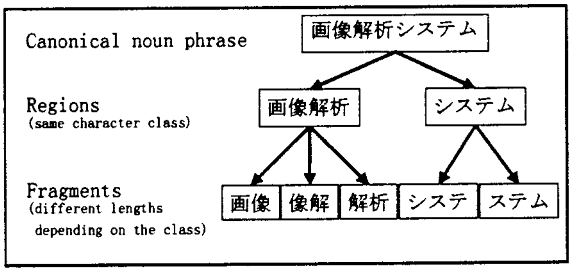
5.2. Step 2: Fragmenting

At this stage, we have all the Japanese noun phrases of the text, in a canonical form, and we start extracting the essence of the text. A usual way to do this in English is to remove the inflexion of the words, namely stemming. There are some stemming algorithms available for English. Lovins (1968) and Porter (1980) are good examples. Turney (1999) shows that, for a keyphrase extraction algorithm, a sophisticated stemmer might not be necessary. Since English stemmers cannot be applied to Japanese, we usually need help from a morphological analyser to segment the compound words (e.g. '画像解析') into their component words (e.g. '画像', '解析'). However, the use of a morphological analyser is time-consuming despite some advances in the restriction of the structural ambiguity (Muranaka, 1997), and is still prone to errors.

In this application, instead of segmenting the component words, we decided to fragment them. This is much more efficient, because we do not need a morphological analyser, and we believe that we are getting the effect needed.

The fragmentation is straight-forward (Figure 7). First, we divide the canonical phrase into a number of regions that have all the same character class. Then, for each region, we fragment them into a fixed length, which depends on the character class (See PJ_*_frag_length in Table 1).

Figure 7: This graph shows an example of fragmentation.



5.3. Step 3: Scoring

The scoring is a very important part of the algorithm, because this is where the program decides which ideas are important, and it decides how to express those ideas.

5.3.1. Scoring the fragments

Each fragment is assigned a score, which is calculated using the frequency and a factor that varies depending on the position of the first occurrence of the fragment.[2] (See PJ_first_* in Table 1)

After scoring each fragment, we get an ordered list of fragments, which we interpret as a list of the most important concepts and ideas conveyed in the text. The list is of 'PJ_num_working' length (Table 1).

5.3.2. Choosing the best way to express the idea represented by each fragment

For each of the concepts contained in the ordered list of fragments, we expand them to a candidate phrase. For a given concept, we assign a score to each of the possible expansions (candidates), using the frequency, the first occurrence (See PJ_first_* in Table 1), the length of the candidates (See PJ_*_length_* in Table 1), and also the likelihood of the candidates to be a keyphrase (propitious contexts). The choice of the expansion (or, the candidate phrase) for a concept (fragment) is resolved by choosing the candidate with the best score.

In some cases, different concepts can be expanded to the same expression so, if this happens, we remove the duplicates and keep only the expression of the highest scoring fragment.

5.3.3. Apply a different weight to each category of keyphrase

The final part of the scoring process is to re-order the best candidate keyphrases. Because the distribution of the keyphrase's categories is not uniform (Table 3), and also, because the score is calculated differently depending of the category of keyphrase, we cannot use the fragment score as it is, so we re-order by multiplying by different weights assigned to each category of keyphrase. (See PJ_weight_* in Table 1)

5.4. Step 4: Filtering

The final step consists of filtering the verbs identified during the parsing, and removing the candidates which are in a list of stop phrases.

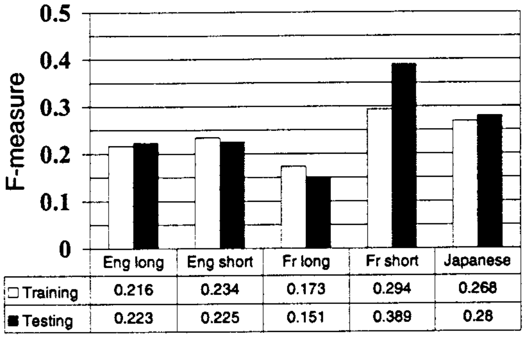
6. Evaluation

In order to evaluate the performance of the algorithm, we used the same method as Turney (1997). We measure the performance in terms of F-measure (Equation 1). The F-measure indicates the degree of overlap between the machine-generated phrases and the author's keyphrases. From Figure 8, we see that we achieve a better score for the Japanese version of Extractor than the English version.

$$Fmeasure = \frac{2 \cdot NbKeyMatch}{NbKeyAuthor + NbKeyMachine} \quad (1)$$

[2] Even the fragments of the "sahen-verbs" are used for the frequency count. Ex.: "画像解析" contains one compound that is the same as one of "解析する".

Figure 8: This graph shows the value of the F-measure depending on the language. (Corpus: "Eng. long" = long English documents; "Eng. short" = short English documents; "Fr long" = long French documents; "Fr short" = short French documents; "Japanese" = Japanese documents (long and short)).



	Eng long	Eng short	Fr long	Fr short	Japanese
□ Training	0.216	0.234	0.173	0.294	0.268
■ Testing	0.223	0.225	0.151	0.389	0.28

7. Future work

We chose not to use a morphological analyser during the initial parsing, for the sake of speed and size of the software. However, it would be interesting to see if we could achieve better performance by using morphological analysis.

We are currently in the process of adding automatic selection of highlights to the Japanese version of Extractor. Also, Extractor's support for other languages (German, Korean and Spanish) is currently under development.

8. Conclusion

We presented a solution to the problem of extracting keyphrases from a Japanese document. One limitation of using this approach is that we cannot extract keyphrases that contain hiragana characters. However, since the hiragana keyphrases or the keyphrases that contain some hiragana are very rare (Table 3), this algorithm is sufficient for the vast majority of the documents available. Also, if we would permit hiragana characters to be part of a keyphrase, the effectiveness would be reduced, since it is likely that we would increase the number of errors by allowing them. The choice of using the parsing technique we described in Section 5.1 instead of a complex morphological analyser makes this an efficient keyphrase extraction tool.

Acknowledgements

Thanks to Dr. Peter Turney for his close collaboration on this project.

References

Kano, N. (1995). *Developing International Software for Windows 95 and Windows NT.* Microsoft Press.

Lovins, J.B. (1968). Development of a stemming algorithm, *Mechanical Translation and Computational Linguistics*, 11, 22-31.

Lunde, K. (1993). *Understanding Japanese Information Processing*, O'Reilly & Associates.

Makino, S. and Tsutsui, M. (1996). *A Dictionary of Basic Japanese Grammar*, The Japan Times Ltd.

Makino, S. and Tsutsui, M. (1996). *A Dictionary of Intermediate Japanese Grammar*, The Japan Times Ltd.

Muranaka, N. (1997). "日本語複語名詞構造解析における構造的曖昧さの抑止機構 (A method to reduce the structural ambiguity of Japanese compound noun during analysis) " (in Japanese), Natural Language Processing Laboratory, Niigata University. (http://nlp.info.eng.niigata-u.ac.jp/~mura/work/work.html)

Ogawa, Y., Bessho, A. and Hirose, M. (1993). Simple word strings as compound keywords: An indexing and ranking method for Japanese texts, *SIGIR 93: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Pittsburgh, PA USA, June 27-July 1, 1993. 227-236.

Ogawa, Y. and Matsuda, T. (1997). Overlapping statistical word indexing: A new indexing method for Japanese text, *SIGIR 97*, Philadelphia PA, USA, July 27-July 31, 1997, 226-234.

Porter, M.F. (1980). An algorithm for suffix stripping. *Program; Automated Library and Information Systems*, 14 (3), 130-137.

Turney, P.D. (1997). *Extraction of Keyphrases from Text: Evaluation of Four Algorithms*, NRC Technical Report ERB-1051, National Research Council Canada.

Turney, P.D. (1999). *Learning to Extract Keyphrases from Text*, NRC Technical Report ERB-1057, National Research Council Canada.