Multi-Language Named-Entity Recognition System based on HMM

Kuniko SAITO and Masaaki NAGATA

NTT Cyber Space Laboratories, NTT Corporation

1-1 Hikari-no-oka Yokosuka-Shi Kanagawa, 239-0847 Japan

{saito.kuniko,nagata.masaaki}@lab.ntt.co.jp

Abstract

We introduce a multi-language named-entity recognition system based on HMM. Japanese, Chinese, Korean and English versions have already been implemented. In principle, it can analyze any other language if we have training data of the target language. This system has a common analytical engine and it can handle any language simply by changing the lexical analysis rules and statistical language model. In this paper, we describe the architecture and accuracy of the named-entity system, and report preliminary experiments on automatic bilingual named-entity dictionary construction using the Japanese and English named-entity recognizer.

1. Introduction

There is increasing demand for cross-language information retrieval. Due to the development of the World Wide Web, we can access information written in not only our mother language but also foreign languages. One report has English as the dominant language of web pages (76.6 %), followed by Japanese (2.77 %), German (2.28 %), Chinese (1.69 %), French (1.09 %), Spanish (0.81 %), and Korean (0.65 %) [1]. Internet users who are not fluent in English finds this situation far from satisfactory; the many useful information sources in English are not open to them.

To implement a multi-language information retrieval system, it is indispensable to develop multi-language text analysis techniques such as morphological analysis and named-entity recognition. They are needed in many natural language processing applications such as machine translation, information retrieval, and information extraction.

We developed a multi-language named-entity recognition system based on HMM. This system is mainly for Japanese, Chinese, Korean and English, but it can handle any other language if we have training data of the target language. This system has a common analytical engine and only the lexical analysis rules and statistical language model need be changed to handle any other language. Previous works on multi-language named-entity recognition are mainly for European languages [2]. Our system is the first one that can handle Asian languages, as far as we know.

In the following sections, we first describe the system architecture and language model of our named-entity recognition system. We then describe the evaluation results of our system. Finally, we report preliminary experiments on the automatic construction of a bilingual named-entity dictionary.

2. System Architecture

Our goal is to build a practical multi-language named-entity recognition system for multi-language information retrieval. To accomplish our aim, there are several conditions that should be fulfilled. First is to solve the differences between the features of languages. Second is to have a good adaptability to a variety of genres because there are an endless variety of texts on the WWW. Third is to combine high accuracy and processing speed because the users of information retrieval are sensitive to processing speed. To fulfill the first condition, we divided our system architecture into language dependent parts and language independent parts. For the second and third conditions, we used a combination of statistical language model and optimal word sequence search. Details of the language model and word sequence search are discussed in more depth later; we start with an explanation of the system's architecture.

Figure 1 overviews the multi-language named-entity recognition system. We have implemented Japanese (JP), Chinese (CN), Korean (KR) and English (EN) versions, but it can, in principle, treat any other language.

There are two language dependent aspects. One involves the character encoding system, and the other involves the language features themselves such as orthography, the kinds of character types, and word segmentation. We adopted a character code converter for the former and a lexical analyzer for the latter.

In order to handle language independent aspects, we adopted N-best word sequence search and a statistical language model in the analytical engine.

The following sections describe the character code converter, lexical analyzer, and analytical engine.

2.1. Character Code Conversion

If computers are to handle multilingual text, it is essential to decide the character set and its encoding. The character set is a collection of characters and encoding is a mapping between numbers and characters. One character set could have several encoding schemes. Hundreds of character sets and attendant encoding schemes are used on a regional basis. Most of them are standards from the countries where the language is spoken, and differ from country to country. Examples include JIS from Japan, GB from China and KSC from Korea; EUC-JP, EUC-CN and EUC-KR are the corresponding encoding schemes [3]. We call these encoding schemes 'local codes' in this paper. It is impossible for local code to handle two different character sets at the same time, so Unicode was invented to bring together all the languages of the world [4]. In Unicode, character type is defined as Unicode property through the assignment of a range of code points such as alphanumerics, symbols, kanji (Chinese character), hiragana (Japanese syllabary character), hangul (Korean character) and so on. The proposed lexical analyzer allows us to define arbitrary properties other than those defined by the Unicode standard.

The character code converter changes the input text encoding from local code to Unicode and the output from Unicode to local code. That is, the internal code of our system is Unicode (UCS-4). Our system can accept EUC-JP, EUC-CN, EUC-KR and UTF-8 as input-output encoding schemes. In principle, we can use any encoding scheme if the encoding has round-trip conversion mapping between Unicode. We assume that the input encoding is either specified by the user, or automatically detected by using conventional techniques such as [5].

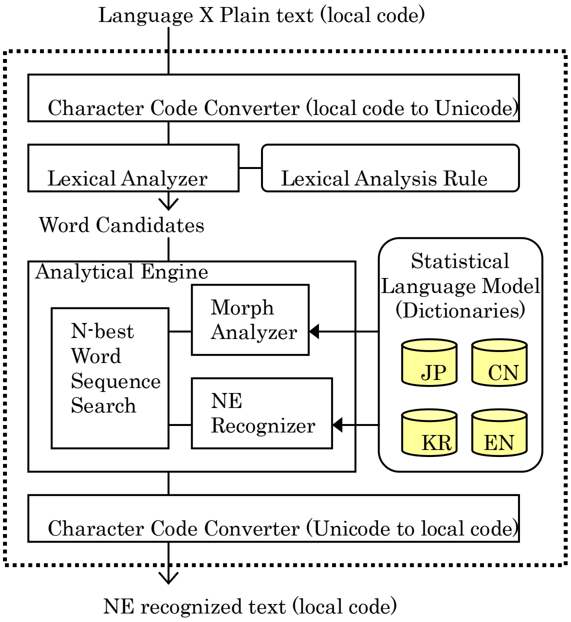
Language X Plain text (local code)
Character Code Converter (local code to Unicode)
Lexical Analyzer
Lexical Analysis Rule
Word Candidates
Analytical Engine
N-best Word Sequence Search
Morph Analyzer
NE Recognizer
Statistical Language Model (Dictionaries)
JP
CN
KR
EN
Character Code Converter (Unicode to local code)
NE recognized text (local code)

Figure 1. System Overview

2.2. Lexical Analyzer

The lexical analyzer recognizes words in the input sentence. It also plays an important role in solving the language differences, that is, it generates adequate word candidates for every language.

The lexical analyzer uses regular expressions and is controlled by lexical analysis rules that reflect the differences in language features. We assume the following three language features;

1. character type and word length
2. orthography and spacing
3. word candidate generation

The features can be set as parameters in the lexical analyzer. We explain these three features in the following sections.

2.2.1 Character Type and Word Length

Table 1 shows the varieties of character types in each language. Character types influence the average word length. For example, in Japanese, kanji (Chinese character) words have about 2 characters and katakana (phonetic character used primarily to represent loanwords) words are about 5 characters long such as ‘パスワード (password)’. In Chinese, most kanji words have 2 characters but proper nouns for native Chinese are usually 3 characters, and those representing loanwords are about 4 characters long such as ‘贝克汉姆 (Beckham)’. In Korean, one hangul corresponds to one kanji and one hangul consists of one consonant - one vowel - one consonant, so loanwords written in hangul are about 3 characters long such as ‘인터넷 (internet)’. Character type and word length are related to word candidate generation in section 2.2.3.

Table 1. Character Types

EN	alphabet symbol number
JP	alphabet symbol number kanji hiragana katakana
CN	alphabet symbol number kanji
KR	alphabet symbol number kanji hangul

2.2.2 Orthography and Spacing

There is an obvious difference in orthography between each language, that is, European languages put a space between words while Japanese and Chinese do not. In Korean, spaces are used to delimit phrases (called as eojeol in Korean) not words, and space usage depends greatly on the individual.

Therefore, another important role of the lexical analyzer is to handle spaces. In Japanese and Chinese, spaces should usually be recognized as tokens, but in English and Korean, spaces must be ignored because it indicates words or phrases. For example, the following analysis results are preferred;

I have a pen →

‘I/pronoun’ ‘have/verb’ ‘a/article’ ‘pen/noun’

and never must be analyzed as follows;

‘I/pronoun’ ‘ /space’ ‘have/verb’ ‘ /space’
‘a/article’ ‘ /space’ ‘pen/noun’

There are, however, many compound nouns that include spaces such as ‘New York’, ‘United States’ and so on. In this case, spaces must be recognized as a character in a compound word. In Korean, it is necessary not only to segment one phrase separated by a space like Japanese, but also to recognize compound words including spaces like English.

These differences in handling spaces are related to the problem of whether spaces must be included in the statistical language model or not. In Japanese and Chinese, it is rare for spaces to appear in a sentence, so the appearance of a space is an important clue in improving analysis accuracy. In English and Korean, however, they are used so often that they don’t have any important meaning in the contextual sense.

The lexical analyzer can treat spaces appropriately. The rules for Japanese and Chinese, always recognize a space as a token, while for those for English and Korean consider spaces only a part of compound words such as ‘New York’.

2.2.3 Word Candidate Generation

In our system, the analytical engine can list all dictionary word candidates from the input string by dictionary lookup. However, it is also necessary to generate word candidates for other than dictionary words, i.e. unknown words candidates. We use the lexical analyzer to generate word candidates that are not in the dictionary.

It is more difficult to generate word candidates for Asian languages than for European languages, because Asian languages don’t put a space between words as mentioned above.

The first step in word candidate generation is to make word candidates from the input string. The simplest way is to list all substrings as word candidates at every point in the sentence. This technique can be used for any language but its disadvantage is that there are so many linguistically meaningless candidates that it takes too long to calculate the probabilities of all combinations of the candidates in the following analytical process. A much more effective approach is to limit word candidates to only those substrings that are likely to be words.

The character types are often helpful in word candidate generation. For example, a cross-linguistic characteristic is that numbers and symbols are often used for serial numbers, phone numbers, block numbers, and so on, and some distinctive character strings of alphabets and symbols such as 'http://www…' and 'name@abc.mail.address' are URLs, Email-addresses and so on. This is not foolproof since the writing styles often differ from language to language. Furthermore, it is better to generate such kinds of word candidates based on the longest match method because substrings of these candidates do not usually constitute a word.

In Japanese, a change between character types often indicates a word boundary. For example, katakana words are loanwords and so must be generated based on the longest match method. In Chinese and Korean, sentences mainly consist of one character type, such as kanji or hangul, so the character types are not as effective for word recognition as they are in Japanese. However, changes from kanji or hangul to alphanumerics and symbols often indicate word changes.

And word length is also useful to put a limit on the length of word candidates. It is a waste of time to make long kanji words (length is 5 or more characters) in Japanese unless the substring matched with the dictionary, because its average length is about 2 characters. In Korean, although hanguls (syllabaries) are converted into a sequence of hangul Jamo (consonant or vowel) internally in order to facilitate the morphological analysis, the length of hangul words are defined in hangul syllabaries.

We designed the lexical analyzer so that it can correctly treat spaces and word candidate generation depending on the character types for each language. Table 2 shows sample lexical analysis rules for Japanese (JP) and English (EN). For example, in Japanese, if character type is kanji or hiragana, the lexical analyzer attempts to output word candidates with lengths of 1 to 3. If character type is katakana, alphabet, or number, it generates one candidate based on the longest match method until character type changes. If the input is '1500km', word candidates are '1500' and 'km'. Subset character strings such as '1', '15', '500', 'k' and 'm' are never output as candidates. It is possible for a candidate to consist of several character types. Japanese has many words that consist of kanji and hiragana such as '離れて(away from)'. In any language there are many words that consist of numbers and alphabetic characters such as '2nd', or alphabetic characters and symbols such as 'U.N.'. Furthermore, if we want to treat positional notation and decimal numbers, we may need to change the Unicode properties, that is, we add '.' and ',' to number-property. The character type 'compound' in English rule indicates compound words. The lexical analyzer generates a compound word (up to 2 words long) with recognition of the space between them. In Japanese, a space is always recognized as one word, a symbol.

Table 3 shows the word candidates output by the lexical analyzer following the rules of Table 2. The Japanese and English inputs are parallel sentences. It is apparent that the efficiency of word candidate generation improves dramatically compared to the case of generating all character strings as

Table 2. Lexical Analysis Rule

	Character Type	Word Length
JP	kanji	1-3
	hiragana	1-3
	katakana	until type changes
	alphabet	until type changes
	number	until type changes
	symbol	1
	kanji – hiragana	1-3
EN	alphabet	until type changes
	number	until type changes
	symbol	1
	compound	up to 2 words

candidates at every point in a sentence. In Japanese, kanji and hiragana strings become several candidates with lengths of 1 to 3, and alphabet and katakana strings become one candidate based on the longest match method until character type changes. In English, single words and compound words are recognized as candidates. Only the candidates that are not in the dictionary become unknown word candidates in the analytical engine.

2.3. Analytical engine

The analytical engine consists of N-best word sequence search and a statistical language model. Our system uses a word bigram model for morphological analysis and a hidden Markov model for named-entity recognition. These models are trained from tagged corpora that have been manually word segmented, part-of-speech tagged, and named-entity recognized respectively. Since N-best word sequence search and statistical language model don’t depend on language, we can apply this analytical engine to all languages. This makes it possible to treat any language if a corpus is available for training the language model. The next section explains the hidden Markov model used for named-entity recognition.

3. Named-entity Recognition Model

The named-entity task is to recognize entities such as organizations, personal names, and locations. Several papers have tackled named-entity recognition through the use of Markov model (HMM) [6], maximum entropy method (ME) [7, 8], and support vector machine (SVM) [9]. It is generally said that HMM is inferior to ME and SVM in terms of accuracy, but is superior with regard to training and processing speed. That is, HMM is suitable for applications that require realtime response or have to process large amounts of text such as information retrieval. We extended the original HMM reported by BBN. BBN’s named-entity system is for English and offers high accuracy.

Table 3. Outputs of Lexical Analyzer

Input sentence	
東京ディズニーランドは、東京駅から10km離れている	Tokyo Disneyland is 10 km away from the Tokyo station.
Word Candidates	
東 東京 ディズニーランド は 、 東　東京　東京駅 京　京駅　京駅か 駅　駅か　駅から か　から ら 10 km 離　離れ　離れて れ　れて　れてい て　てい　ている い　いる る	‘Tokyo’ ‘Tokyo Disneyland’ ‘Disneyland’ ‘Disneyland is’ ‘is’ ‘10’ ‘km’ ‘km away’ ‘away’ ‘away from’ ‘from’ ‘from the’ ‘the’ ‘the Tokyo’ ‘Tokyo’ ‘Tokyo station’ ‘station’ ‘.’

The HMM used in BBN's system is described as follows. Let the morpheme sequence be $W = w_1 \cdots w_n$ and Name Class (NC) sequence be $NC = NC_1 \cdots NC_n$. Here, NC represents the type of named entity such as organization, personal name, or location. The joint probability of word sequence and NC sequence $P(W, NC) = \prod P(w_i, NC_i)$ are calculated as follows;

(1) if $NC_i \neq NC_{i-1}$

$$P(w_i, NC_i) = P(NC_i \mid NC_{i-1}, w_{i-1}) \times P(w_i \mid NC_i, NC_{i-1})$$

(2) if $NC_i = NC_{i-1}$ and $NC_i = NC_{i+1}$

$$P(w_i, NC_i) = P(w_i \mid w_{i-1}, NC_i)$$

(3) if $NC_i = NC_{i-1}$ and $NC_i \neq NC_{i+1}$

$$P(w_i, NC_i) = P(w_i \mid w_{i-1}, NC_i) \times P(< end > \mid w_i, NC_i)$$

Here, the special symbol $< end >$ indicates the end of an NC sequence.

In this model, morphological analysis and named-entity recognition can be performed at the same time. This is preferable for Asian languages because they have some ambiguity about word segmentation. To adapt BBN’s HMM for Asian languages, we extended the original HMM as follows. Due to the ambiguity of word segmentation, morphological analysis is performed prior to applying the HMM; the analysis uses a word bigram model and N-best candidates (of morpheme sequence) are output as a word graph structure. Named-entity recognition is then performed over this word graph using the HMM. We use a forward-DP backward-A* N-best search algorithm to get N-best morpheme sequence candidates [10]. In this way, multiple morpheme candidates are considered in named-entity recognition and the problem of word segmentation ambiguity is mitigated.

BBN's original HMM used a back-off model and smoothing to avoid the sparse data problem. We changed this smoothing to linear interpolation to improve the accuracy, and in addition, we used not only the morpheme frequency of terms but also part of speech frequency. Table 4 shows the linear interpolation scheme used here. Underlined items are added in our model. The weight for each probability was decided from experiments.

4. Experiments

To evaluate our system, we prepared original corpora for Japanese, Chinese, Korean and English. The material was mainly taken from newspapers and Web texts. We used the morpheme analysis definition of Pen Tree Bank for English [11], Jtag for Japanese [12], Beijing Univ. for Chinese [13] and MATEC99 for Korean [14]. The named-entity tag definitions were based on MUC [15] for English and IREX [16] for Japanese. We defined Chinese and Korean named-entity tags following the Japanese IREX specifications. Table 5 shows dictionary and corpus size. Dictionary words means the size of the dictionary for morphological analysis. Total words and sentences represent the size of the corpus for named-entity recognition.

Named-entity accuracy is expressed in terms of recall and precision. We also use the F-measure to indicate the overall performance. It is calculated as follows;

$$F = \frac{2 \times recall \times precision}{recall + precision} \quad (4)$$

Table 6 shows the F-measure for all languages. Since we used our original corpora in this evaluation, we cannot compare our results to those of previous works. Accordingly, we also evaluated SVM using our original corpora (see Table 6) [17]. The accuracy of HMM and SVM were approximately equivalent. But the analysis speed of HMM was ten times faster than that of SVM [9]. This means that our system is very fast and has state-of-the-art accuracy in four languages.

We noted that the accuracy of SVM is unusually lower than that of HMM for Japanese. We have not yet confirmed the cause of this, but a plausible argument is as follows. First, the word segmentation ambiguity has a worse affect on accuracy than expected. Since current SVM implementations can not handle N-best morpheme candidates and lower-order candidates are not considered in named-entity recognition. Second, SVM may not suit the analysis of irregular, ill-structured, and informal sentences such as Web texts. Our original corpus data was

Table 4. Linear Interpolation Scheme

$P(NC_i \mid NC_{i-1}, w_{i-1})$	$P(w_i \mid NC_1, NC_{i-1})$	$P(w_i \mid w_{i-1}, NC_1)$
$\underline{P(NC_i \mid NC_{i-1}, pos_{i-1})}$	$\underline{P(pos_i \mid NC_1, NC_{i-1})}$	$\underline{P(pos_i \mid pos_{i-1}, NC_1)}$
$P(NC_i \mid NC_{i-1})$	$P(w_i \mid NC_1)$	$P(w_i \mid NC_1)$
$P(NC)$	$\underline{P(pos_i \mid NC_1)}$	$\underline{P(pos_i \mid NC_1)}$
$1/number\ of\ NC$		

Table 5. Dictionary and Corpus Size

	dictionary words	total words	sentences
EN	17,546	144,708	5,921
JP	436,157	143,408	4,793
CN	147,585	410,188	12,824
KR	182,523	1,456,130	39,943

Table 6. Named Entity Accuracy (F-measure(%))

	HMM	SVM
EN	88.2	84.7
JP	81.0	57.3
CN	84.5	89.5
KR	79.9	82.1

taken from newspapers and Web texts, the former contains complete and grammatical sentences unlike the latter. It is often said that HMM is robust enough to analyze these dirty sentences. It is, anyhow, our next step to analyze the results of named-entity recognition in more detail.

5. Application to Bilingual Lexicon Extraction from Parallel Text

In order to illustrate the benefit of our multi-language named-entity recognition system, we conducted a simple experiment on extracting bilingual named-entity lexicons from parallel texts. It is very difficult to gather bilingual lexicons of named entities because there are an enormous number of new named entities. Establishing a bilingual named-entity dictionary automatically would be extremely useful.

There are 3 steps in extracting a bilingual lexicon as follows;

1. recognize named entity from parallel text
2. extract bilingual lexicon candidates
3. winnow the candidates to yield a reasonable lexicon list

The multi-language named-entity recognition system is used in the first step. In this step, the parallel texts are analyzed morphologically and named entities are recognized.

In the second step, bilingual lexicon candidates are listed automatically under the following conditions;

・word sequence up to 5 words
・include one or more named entities
・does not include function words

The cooccurrence frequency of candidates is calculated at the same time.

In the third step, reasonable lexicons are created from the candidates. To judge the suitability of the candidates to be entered into a bilingual lexicon, we use the translation model called the IBM model [18]. Let a word sequence in language X be $X = x_1 \cdots x_l$ and let the corresponding word sequence in language Y be $Y = y_1 \cdots y_m$. Here, $x_i (1 \le i \le l)$ and $y_j (1 \le j \le m)$ represent one word. In IBM model 1, the conditional probability $P(Y \mid X)$ is calculated as follows;

$$P(Y \mid X) = \frac{\varepsilon}{(l+1)^m} \prod_{j=1}^{m} \cdot \sum_{i=1}^{l} t(y_j \mid x_i) \tag{5}$$

where ε is constant. $t(y_j \mid x_i)$ is translation probability and is estimated by applying the EM algorithm to a large number of parallel texts.

Since the longer word sequences X and Y are, the smaller $P(Y \mid X)$ becomes, the value of $P(Y \mid X)$ cannot be compared when a word sequence length changes. Therefore, we improved equation (5) to take into account the difference in a word sequence length and cooccurrence frequency as follows;

$$S(Y \mid X) = freq \cdot match(X) \cdot match(Y) \cdot \frac{P(Y \mid X)}{E(Y \mid X)} \tag{6}$$

$freq$: cooccurrence frequency of X and Y in parallel text

$match(X)$: ratio of $t(y_j \mid x_i) \ne 0$ in X

$match(Y)$: ratio of $t(y_j \mid x_i) \ne 0$ in Y

$$E(Y \mid X) = \frac{\varepsilon}{(l+1)^m} \prod_{j=1}^{m} \cdot \sum_{i=1}^{l} \bar{t}(y_j \mid x_i)$$

$\bar{t}(y_j \mid x_i)$ is the average of $t(y_j \mid x_i)$. $S(Y \mid X)$ is used as a measure of candidate suitability.

We used Japanese-English news article alignment data as parallel texts that is released by CRL [19, 20]. In this data, articles and sentences are aligned automatically. We separated the parallel text into a small set (about 1000 sentences) and a

Table 7. List of Bilingual Lexicons

North Korea	北朝鮮
United States	米国
International Monetary Fund	国際通貨基金
Soviet Union	ソ連
Middle East	中東
North Atlantic Treaty Organization	北大西洋条約機構
U.S. President Bill Clinton	クリントン米大統領
North American Free Trade Agreement	北米自由貿易協定
European Community	欧州共同体
Taiwan Strait	台湾海峡
Clinton administration	クリントン政権
U.N. General Assembly	国連総会
Tokyo Stock Exchange	東京証券取引所

large set (about 150 thousand sentences). We extracted bilingual lexicons from a small set and $t(y_j \mid x_i)$ was estimated from a large set. Table 7 shows bilingual lexicons that achieved very high scores. It can be said that they are adequate as bilingual lexicons. Though a more detailed evaluation is a future task, the accuracy is about 86 % for the top 50 candidates. This suggests that the proposed system can be applied to bilingual lexicon extraction for automatically creating bilingual dictionaries of named entities.

Conclusion

We developed a multi-language named-entity recognition system based on HMM. We have implemented Japanese, Chinese, Korean and English versions, but in principle it can handle any language if we have training data for the target language. Our system is very fast and has state-of-the-art accuracy.

References

[1] Google: 1.6 Billion Served. *Wired*, December 2000, pp.118-119 (2000).

[2] Cucerzan, S. and Yarowsky, D.: Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence, *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Method in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, College Park, pp. 90-99 (1999)

[3] Lunde, K.: CJKV Information Processing, *O'REILY*, (1999).

[4] The Unicode Consortium.: The Unicode Standard, version 3.0, *Addison-Wesley Longman*, (2000).

[5] Kikui, G.: Identifying the Coding System and Language of On-line Documents on the Internet, *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pp. 652–657 (1996)

[6] Bikel, D. M., Schwartz, R. and Weischedel, R. M.: An Algorithm that Learns What's in a Name, *Machine Learning*, Vol. 34, No. 1-3, pp. 211-231 (1999)

[7] Borthwick, A., Sterling, J., Agichtein, E. and Grishman, R.: Exploiting Diverse Knowledge Sources via Maximum Entropy, *Proceedings of the 6th Workshop on Very Large Corpora* (*VLC-98)*, pp. 152-160 (1998).

[8] Uchimoto, K., Murada, M., Ma, Q., Ozaku, H. and Isahara, H.: Named Entity Extraction Based on A Maximum Entropy Model and Transformation Rules, *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-00)*, pp. 326-335 (2000).

[9] Isozaki, H. and Kazawa, H.: Efficient Support Vector Classifiers for Named Entity Recognition, *Proceedings of the 19th International Conference on Computational Linguistics (COLING-02)*, pp. 390-396 (2002).

[10] Nagata, M.: A Stochastic Japanese Morphological Analyzer Using a Forward-DP Backward-A* N-Best Search Algorithm, *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pp. 201-207 (1994).

[11] Marcus, M. P., Santorini, B. and Marcinkiewicz, M. A.: Building a large annotated corpus of English: The Penn Treebank, *Computational Linguistics*, Vol. 19, No.2, pp. 313-330 (1993).

[12] Fuchi, T. and Takagi, S.: Japanese Morphological Analyzer using Word Co-occurrence -JTAG-, *Proceedings of 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL-COLING-98)*, pp. 409-413 (1998).

[13] Yu, Shiwen. et. al.: The Grammatical Knowledge-base of Contemporary Chinese --- A Complete Specification (现代汉语语法信息词典详解), *Tsinghua University Press*, (1992).

[14] ETRI.: Part-of-Speech Tagset Guidebook 품사 태그 세트 지침서), Unpublished Manual, (1999)

[15] DARPA: *Proceedings of the 7th Message Understanding Conference (MUC-7)* (1998).

[16] IREX Committee (ed.), 1999. *Proceedings of the IREX workshop*. http://nlp.cs.nyu.edu/irex/

[17] Kudo, T and Matsumoto, Y.: Chunking with Support Vector Machines, *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-01)*, pp. 192-199 (2001).

[18] Brown, P.F., Pietra, S. A. D., Pietra, V. J. D. and Mercer, R. L.: The Mathematics of Statistical Machine Translation: Parameter Estimation, *Computational Linguistics*, Vol. 19, No. 2, pp. 263-311 (1993)

[19] Utiyama, M. and Isahara, H.: Reliable Measures for Aligning Japanese-English News Article and Sentences, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03) (2003).*

[20] Japanese-English News Article Alignment Data, http://www2.crl.go.jp/jt/a132/members/mutiyama/jea/index.html (2003)