Language Independent Morphological Analysis

Yamashita, Tatsuo and Matsumoto, Yuji
Graduate School of Information Science
Nara Institute of Science and Technology
{*tatuo-y,matsu*}*@is.aist-nara.ac.jp*

Abstract

This paper proposes a framework of language independent morphological analysis and mainly concentrate on tokenization, the first process of morphological analysis. Although tokenization is usually not regarded as a difficult task in most segmented languages such as English, there are a number of problems in achieving precise treatment of lexical entries. We first introduce the concept of morpho-fragments, which are intermediate units between characters and lexical entries. We describe our approach to resolve problems arising in tokenization so as to attain a language independent morphological analyzer.

1 Introduction

The first step in natural language processing is to identify words in a sentence. We call this process a morphological analysis. Various languages exist in the world, and strategies for morphological analysis differ by types of language. Conventionally, morphological analyzers have been developed in *one analyzer for each language* approach. This is a language dependent approach. In contrast, We propose a framework of language independent morphological analysis system. We employ *one analyzer for any language* approach. This approach enables a rapid implementation of morphological analysis systems for new languages.

We define two types of written languages: one is a *segmented language*, and the other is a *non-segmented language*. In non-segmented languages such as Chinese and Japanese, since words are not separated by delimiters such as white spaces, tokenization is a important and difficult task. In segmented languages such as English, since words are seemingly separated by white spaces or punctuation marks, tokenization is regarded as a relatively easy task and little attention has been paid to. Therefore, each language dependent morphological analyzer has its own strategy for tokenization. We call a string defined in the dictionary *lexeme*. From an algorithmic point of view, *tokenization* is regarded as the process of converting an input stream of characters into a stream of lexemes.

We assume that a morphological analysis consists of three processes: tokenization, dictionary look-up, and disambiguation. Dictionary look-up gets a string and returns a set of lexemes with part-of-speech information. This implicitly contains lemmatization. Disambiguation selects the most plausible sequence of lexemes by a use of a rule-base model or a hidden Markov model (HMM)(Manning and Schütze, 1999). Disambiguation is already language independent, since it does not process strings directly and therefore will not be taken up. On the other hand, tokenization and dictionary look-up are language dependent and shall be explained more in this paper.

We consider problems concerning tokenization of segmented languages in Section 2. To resolve these problem, we first apply the method of non-segmented languages processing to segmented languages (Section 3). However, we do not obtain a satisfactory result. Then, we introduce the concept of morpho-fragments to generalize the method of non-segmented language processing (Section 4). The proposed framework resolves most problems in tokenization, and an efficient language independent part-of-speech tagging becomes possible.

2 Problems of Tokenization in Segmented Languages

In segmented languages such as English, tokenization is regarded as a relatively easy task and little attention has been paid to. When a sentence has clear word boundaries, the analyzer just consults the dictionary look-up component whether strings between delimiters exist in the dictionary. If any string exists, the dictionary look-up component returns the set of possible parts-of-speech. This string is known as *graphic word* which is defined as "a string of contiguous alphanumeric characters with space on either side; may include hyphens and apostrophes, but no other punctuation marks" (Kučera and Francis, 1967).

Conventionally, in segmented languages, an analyzer converts a stream of characters into graphic words (see the rows labeled "Characters" and

Sentence	Dr. Lee and John's son go to the McDonald's in New York.
Characters	|D|r|.| |L|e|e| |a|n|d| |J|o|h|n|'|s| |s|o|n| |g|o| |t|o| |t|h|e| |M|c|D|o|n|a|l|d|'|s| |i|n| |N|e|w| |Y|o|r|k|.|
Graphic Words	|Dr|.| |Lee| |and| |John's| |son| |go| |to| |the| |McDonald's| |in| |New| |York|.|
Lexemes	|Dr.| |Lee| |and| |John|'s| |son| |go| |to| |the| |McDonald's| |in| |New York|.|
Morpho-fragments	|Dr|.| |Lee| |and| |John|'|s| |son| |go| |to| |the| |McDonald|'|s| |in| |New| |York|.|

Figure 1: Decomposition of Sentence in English

Sentence	彼は休暇で帰省している。 (He is home for holiday.)
Characters	| 彼 | は | 休 | 暇 | で | 帰 | 省 | し | て | い | る | 。
Lexemes	| 彼 | は | 休暇 | で | 帰省 | して | いる | 。
Morpho-fragments	| 彼 | は | 休 | 暇 | で | 帰 | 省 | し | て | い | る | 。

Figure 2: Decomposition of Sentence in Japanese

"Graphic Words" in Figure 1) and searches the dictionary for these graphic words. However, in practice, we want a sequence of lexemes (see the line labeled "Lexemes" in Figure 1). We list two major problems of tokenization in segmented languages below (examples in English). We use the term *segment* to refer to a string separated by white spaces.

1. Segmentation(one segment into several lexemes):

   Segments with a period at the end (e.g, "Calif." and "etc.") suffer from segmentation ambiguity. The period can denote an abbreviation, the end of a sentence, or both. The problem of sentence boundary ambiguity is not easy to solve (Palmer and Hearst, 1997). A segment with an apostrophe also has segmentation ambiguity. For example, "McDonald's" is ambiguous since this string can be segmented into either "McDonald / Proper noun" + " 's / Possessive ending" or "McDonald's / Proper noun (company name)". In addition, "boys' " in a sentence "... the boys' toys ..." is ambiguous. The string can be segmented into either "boys' / Plural possessive" or "boys / Plural Noun" + " ' / Punctuation (the end of a quotation)" (Manning and Schütze, 1999). If a hyphenated segment such as "data-base," "F-16," or "MS-DOS" exists in the dictionary, it should be an independent lexeme. However, if a hyphenated segment such as "55-years-old" does not exist in the dictionary, hyphens should be treated as independent tokens(Fox, 1992). Other punctuation marks such as "/" or "_" have the same problem in "OS/2" or "max_size"(in programming languages).

2. Round-up(several segments into one lexeme):

   If a lexeme consisting of a sequence of segments such as a proper noun (e.g., "New York") or a phrasal verb (e.g., "look at" and "get up") exists in the dictionary, it should be a lexeme. To handle such lexemes, we need to store multi-segment lexemes in the dictionary. Webster and Kit handle idioms and fixed expressions in this way(Webster and Kit, 1992). In Penn Treebank(Santorini, 1990), a proper noun like "New York" is defined as two individual proper nouns "New / NNP" + "York / NNP," disregarding round-up of several segments into a lexeme.

The definition of lexemes in a dictionary depends on the requirement of application. Therefore, a simple pattern matcher is not enough to deal with language independent tokenization.

Non-segmented languages do not have a delimiter between lexemes (Figure 2). Therefore, a treatment of further segmentation and rounding up has been well considered. In a non-segmented language, the analyzer considers all prefixes from each position in the sentence, checks whether each prefix matches the lexeme in the dictionary, stores these lexemes in a graph structure, and finds the most plausible sequence of lexemes in the graph structure. To find the sequence, Nagata proposed a probabilistic language model for non-segmented languages(Nagata, 1994)(Nagata, 1999).

The crucial difference between segmented and

non-segmented languages in the process of morphological analysis appears in the way of the dictionary look-up. The standard technique for looking up lexemes in Japanese dictionaries is to use a trie structure(Fredkin, 1960)(Knuth, 1998). A trie structured dictionary gives all possible lexemes that start at a given position in a sentence effectively(Morimoto and Aoe, 1993). We call this method of word looking-up as "common prefix search" (hereafter CPS). Figure 3 shows a part of the trie for Japanese lexeme dictionary. The results of CPS for "海老名へ行く。"(I go to Ebina.) are "海老" and "海老名." To get all possible lexemes in the sentence, the analyzer has to slide the start position for CPS to the right by character by character.

3 A Naive Approach

A simple method that directly applies the morphological analysis method for non-segmented languages can handle the problems of segmented languages. For instance, to analyze the sentence, "They've gone to school together," we first delete all white spaces in the sentence and get "They'vegonetoschooltogether." Then we pass it to the analyzer for non-segmented languages. However, the analyzer may return the result as "They / 've / gone / to / school / to / get / her / ." inducing a spurious ambiguity. Mills applied this method and tokenized the medieval manuscript in Cornish(Mills, 1998).

We carried out experiments to examine the influence of delimiter deletion. We use Penn Treebank(Santorini, 1990) part-of-speech tagged corpus (1.3M lexemes) to train an HMM and analyze sentences by HMM-based morphological analyzer MOZ(Yamashita, 1999)(Ymashita et al., 1999). We use a bigram model for training it from the corpus. Test data is the same as the training corpus. Table 1 shows accuracy of segmentation and part-of-speech tagging. The accuracy is expressed in terms of recall and precision(Nagata, 1999). Let the number of lexemes in the tagged corpus be Std, the number of lexemes in the output of the analyze be Sys, and the number of matched lexemes be M. *Recall* is defined as M/Std, *precision* is defined as M/Sys. The following are the labels in Table 1 (sentence formats and methods we use):

LXS We isolate all the lexemes in sentences and apply the method for segmented languages to the sentences. This situation is ideal, since the problems we discussed in Section 2 do not exist. In other words, all the sentences do not have segmentation ambiguity. We use the results as the baseline. Example sentence: "`It␣'s␣Mr.␣Lee␣'s␣pen␣.`"

NSP We remove all the spaces in sentences and apply the method for non-segmented languages to the sentences. Example sentence: "`It'sMr.Lee'spen.`"

NOR Sentences are in the original normal format. We apply the method for non-segmented languages to the sentences. Example sentence: "`It's␣Mr.␣Lee's␣pen.`"

Because of no segmentation ambiguity, "LXS" performs better than "NSP" and "NOR." The following are typical example of segmentation errors. The errors originate from conjunctive ambiguity and disjunctive ambiguity(Guo, 1997).

conjunctive ambiguity The analyzer recognized "away," "ahead," "anymore," and "workforce" as "a way," "a head," "any more," and "work force," respectively. In the results of "NSP," the number of this type of error is 11,267.

disjunctive ambiguity The analyzer recognized "a tour," "a ton," and "Alaskan or" as "at our," "at on," and "Alaska nor," respectively. In the results of "NSP," the number of this type of error is 233.

Since only "NSP" has disjunctive ambiguity, "NOR" performs better than "NSP." This shows that white spaces between segments help to decrease segmentation ambiguity.

Though the proportional difference in accuracy looks slight between these models, there is a considerable influence in the analysis efficiency. In the cases of "NSP" and "NOR," the analyzer may look up the dictionary from any position in a given sentence, therefore candidates for lexemes increase, and the analysis time also increase. The results of our experiments show that the run time of analyses of "NSP" or "NOR" takes about 4 times more than that of "LXS."

4 Morpho-fragments: The Building Blocks

Although segmented languages seemingly have clear word boundaries, there are problems of further segmentation and rounding up as introduced in Section 2. The naive approach in Section 3 does not work well. In this section, we propose an efficient and sophisticated method to solve the problems by introducing the concept of *morpho-fragments*. We also show that a uniform treatment of segmented and non-segmented languages is possible without inducing the spurious ambiguity.

4.1 Definition

The *morpho-fragments* (MFs) of a language is defined as the smallest set of strings of the alphabet which can compose all lexemes in the dictionary. In other words, MFs are intermediate units between

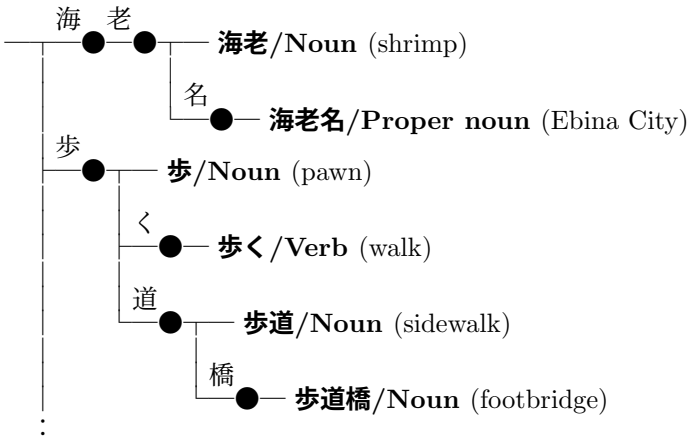

Figure 3: Japanese Trie Structured Dictionary

	Segmentation (Recall / Precision)	POS Tagging (Recall / Precision)	Analysis Time (Ratio)
LXS	100	96.98	1.0
NSP	99.52 / 99.67	96.52 / 96.69	4.3
NOR	99.87 / 99.91	96.84 / 96.88	4.2
MF	99.88 / 99.93	96.85 / 96.91	1.4

Table 1: Results of Experiments

characters and lexemes (see Figure 1 and Figure 2). MFs are well defined tokens which are specialized for language independent morphological analysis.

For example, in English, all punctuation marks are MFs. Parts of a token separated by a punctuation mark such as "He," "s," and the punctuation mark itself, " ' " in "He's" are MFs. The tokens in a compound lexeme such as "New" and "York" in "New York" are also MFs. In non-segmented languages such as Chinese and Japanese, every single character is a MF. Figure 4 shows decomposition of sentences into MFs (enclosed by "[" and "]") for several languages. *Delimiters* (denoted "␣") are treated as special MFs that cannot start nor end a lexeme.

Once the set of MFs is determined, the dictionary is compiled into a trie structure in which the edges are labeled by MFs, as shown in Figure 5 for English and in Figure 3 for Japanese. A trie structure ensures to return all and only possible lexemes starting at a particular position in a sentence by a one-time consultation to the dictionary, resulting in an efficient dictionary look-up with no spurious ambiguity.

When we analyze a sentence of a non-segmented language, to get all possible lexemes in the sentence, the analyzer slides the position one character by one character from the beginning to the end of the sentence and consults the trie structured dictionary (Section 2). Note that every character is a MF in non-segmented languages. In the same way, to analyze a sentence of a segmented language, the analyzer slides the position one MF by one MF and consults the trie structured dictionary, then, all possible lexemes are obtained. For example, in Figure 5, the results of CPS for "'m in ..." are "' " and "'m," and the results for "New York is ..." are "New" and "New York."

Therefore, a morphological analyzer with CPS-based dictionary look-up for non-segmented languages can be used for the analysis of segmented languages. In other words, MFs make possible language independent morphological analysis. We can also say MFs specify the positions to start as well as to end the dictionary look-up.

Language	Sentence		Recognized Morpho-fragments
English	I'm in New York.	→	[I][']␣[m]␣[in]␣[New]␣[York][.]
Chinese	他是我弟弟. (He is my little brother.)	→	[他][是][我][弟][弟][.]
Korean	나는 학교에 간다. (I go to school.)	→	[나][는]␣[학][교][에]␣[간][다][.]
Japanese	学校へ行きましょう。(Let's go to school.)	→	[学][校][へ][行][き][ま][し][ょ][う][。]

Figure 4: Recognition of Morpho-fragments

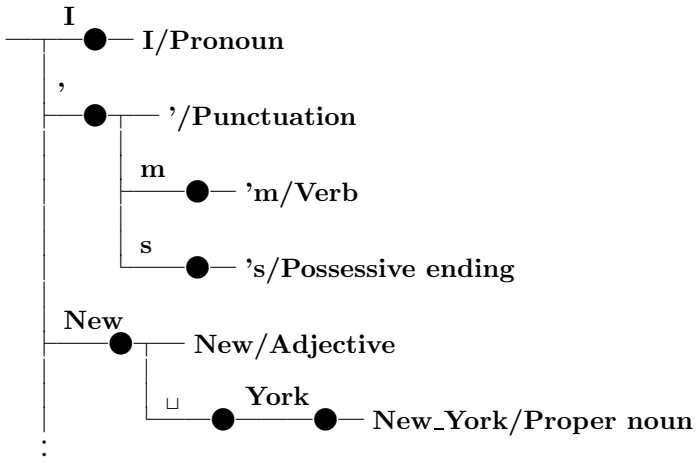

Figure 5: English Trie Structured Dictionary

4.2 How to Recognize Morpho-fragments

The problem is that it is not easy to identify the complete set of MFs for a segmented language. We do not make effort to find out the minimum and complete set of MFs. Instead, we decide to specify all the possible delimiters and punctuation marks appearing in the dictionary, these may separate MFs or become themselves as MFs. By specifying the following three kinds of information for the language under consideration, we attain a pseudo-complete MF definition. The following setting not only simplifies the identification of MFs but also achieves a uniform framework of language dependent morphological analysis system.

1. The language type:
   The languages are classified into two groups: segmented and non-segmented languages. "Language type" decides if every character in the language can be an MF. In non-segmented language every character can be an MF. In segmented language, punctuation marks and sequences of characters except for delimiters can be an MF.
2. The set of the delimiters acting as boundaries:
   These act as boundaries of MFs. However, these can not be independent MFs (can not start nor end a lexeme). For example, white spaces are delimiters in segmented languages.
3. The set of the punctuation marks and other symbols:
   These act as a boundary of MFs as well as an MF. Examples are an apostrophe in "It's," a period in "Mr.," and a hyphen in "F-16."

Using these information, the process of recognizing MFs becomes simple and easy. The process can be implemented by a finite state machine or a simple pattern matcher.

The following is the example of the definition for English:

1. Language type: segmented language
2. Delimiters: white spaces, tabs, and carriage-returns
3. Punctuation marks: [.][,][:][;][']["][-]···[0][1][2]···

As is clear from the definition, "punctuation marks" are not necessary for non-segmented language, since

every character is an MF. The following is the example for Japanese and Chinese.

1. Language type: non-segmented language
2. Delimiters: not required
3. Punctuation marks: not required

Though Korean sentences are separated by spaces into phrasal segments, Korean is a non-segmented language essentially, since each phrasal segment does not have lexeme boundaries. We call this type of languages incompletely-segmented languages. German is also categorized as this type. The following is the example for Korean.

1. Language type: non-segmented language
2. Delimiters: spaces, tabs, and carriage-returns
3. Punctuation marks: not required

In incompletely-segmented languages, such as Korean, we have to consider two types of connection of lexemes, one is "over a delimiter" and the other is "inside a segment" (Hirano and Matsumoto, 1996). If we regard delimiters as lexemes, a trigram model can make it possible to treat both types.

The definition gives possible starting positions of MFs in sentences of the language and the same morphological analysis system is usable for any language.

We examined an effect of applying the morpho-fragments to analysis. Conditions of the experiment are almost the same as "NOR." The difference is that we use the morpho-fragments definition for English. The row labeled "MF" in Table 1 shows the results of the analysis. Using the morpho-fragments decreases the analysis time drastically. The accuracy is also better than those of the naive approaches.

Well studied language such as English may have a hand tuned tokenizer which is superior to ours. However, to tune a tokenizer by hand is not suitable to implement many minor languages.

4.3 Implementation

We implement a language independent morphological analysis system based on the concept of morpho-fragments(Yamashita, 1999). With an existence of tagged corpus, it is straightforward to implement part-of-speech taggers. We have implemented several of such taggers. The system uses an HMM. This is trained by a part-of-speech tagged corpus. We overview the setting and performance of tagging for several languages.

English

An HMM is trained by the part-of-speech tagged corpus part of Penn Treebank(Santorini, 1990) (1.3 million morphemes). We use a trigram model. The lexemes in the dictionary are taken from the corpus as well as from the entry words in Oxford Advanced Learner's Dictionary(Mitton, 1992). The system achieves 97% precision and recall for training data, 95% precision and recall for test data.

Japanese

An HMM is trained by Japanese part-of-speech tagged corpus(Rea, 1998) (0.9 million morphemes). We use a trigram model. The lexemes in the dictionary are taken from the corpus as well as from the dictionary of ChaSen(Matsumoto et al., 1999), a freely available Japanese morphological analyzer. The system achieves 97% precision and recall for training and test data.

Chinese

An HMM is trained by the Chinese part-of-speech tagged corpus released by CKIP(Chinese Knowledge Information Processing Group, 1995) (2.1 million morphemes). We use a bigram model. The lexemes in the dictionary are taken only from the corpus. The system achieves 95% precision and recall for training data, 91% precision and recall for test data.

5 Related Work and Remarks

We address two problems of tokenization in segmented languages: further segmentation and round-up. These problems are discussed by several authors including Mills(Mills, 1998) Webster & Kit(Webster and Kit, 1992). However, their proposed solutions are not language independent.

To resolve the problems of tokenization, we first apply the method of non-segmented languages processing. However, this causes spurious segmentation ambiguity and a considerable influence in the analysis times. Therefore, we propose the concept of morpho-fragments that minimally comprises the lexemes in a language. Although the idea is quite simple, our approach avoids spurious ambiguity and attains an efficient look-up of a trie structured dictionary. In conclusion, the concept of morpho-fragments makes it easy to implemented language independent morphological analysis.

References

Chinese Knowledge Information Processing Group, 1995. **中央研究院平衡語料庫的內容與說明**. Academia Sinica Institute of Information Science. in Chinese.

Christopher Fox, 1992. *Lexical Analysis and Stoplists*, chapter 7, pages 102–130. In Frakes and Baeza-Yates (Frakes and Baeza-Yates, 1992).

William B. Frakes and Ricardo A. Baeza-Yates, editors. 1992. *Information Retrieval: Data Structures & Algorithms*. Prentice-Hall.

Edward Fredkin. 1960. Trie memory. *Communications of the ACM*, 3(9):490–500, September.

Jin Guo. 1997. Critical tokenization and its properties. *Computational Linguistics*, 23(4):569–596, December.

Yoshitaka Hirano and Yuji Matsumoto. 1996. A proposal of korean conjugation system and its application to morphological analysis. In *Proceedings of the 11th Pacific Asia Conference on Language, Information and Computation (PACLIC 11)*, pages 229–236, December.

Donald E. Knuth. 1998. *The Art of Computer Programming : Sorting and Searching*, volume 3. Addison-Wesley, second edition, May.

Henry Kučera and W. Nelson Francis. 1967. *Computational Analysis of Present-Day American English*. Brown University Press.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, and Yoshitaka Hirano, 1999. *Japanese Morphological Analysis System ChaSen version 2.0 Manual*. NAIST Technical Report NAIST-IS-TR99009, April.

Jon Mills. 1998. Lexicon based critical tokenization: An algorithm. In *Euralex'98*, pages 213–220, August.

Roger Mitton, 1992. *A Description of A Computer-Usable Dictionary File Based on The Oxford Advanced Learner's Dictionary of Current English*.

K. Morimoto and J. Aoe. 1993. Two trie structures for natural language dictionaries. In *Proceedings of Natural Language Processing Pacific Rim Symposium*, pages 302–311.

Masaaki Nagata. 1994. A stochastic japanese morphological analyzer using a forward-dp backward-a* n-best search algorithm. In *COLING-94*, volume 1, pages 201–207, August.

Masaaki Nagata. 1999. A part of speech estimation method for japanese unknown words using a statistical model of morphology and context. In *37th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 277–284, June.

David D. Palmer and Marti A. Hearst. 1997. Adaptive multilingual sentence boundary disambiguation. *Computational Linguistics*, 23(2):241–267, June.

Real World Computing Partnership, 1998. *RWC Text Database Report*. in Japanese.

Beatrice Santorini, 1990. *Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision, 2nd printing)*, June.

Jonathan J. Webster and Chunyu Kit. 1992. Tokenization as the initial phase in nlp. In *COLING-92*, volume 4, pages 1106–1110, August.

Tatsuo Yamashita, 1999. *MOZ and LimaTK Manual*. NAIST Computational Linguistics Laboratory, <http://cl.aist-nara.ac.jp/~tatuo-y/ma/>, August. in Japanese.

Tatsuo Ymashita, Msakazu Fujio, and Yuji Matsumoto. 1999. Language independent tools for natural language processing. In *Proceedings of the Eighteenth International Conference on Computer Processing*, pages 237–240, March.