NTT DATA: DESCRIPTION OF THE ERIE SYSTEM USED FOR MUC-6

Yoshio Eriguchi and Tsuyoshi Kitani
NTT Data Communications Systems Corporation
Kowa Kawasaki Nishi-guchi Bldg., 66-2 Horikawa-cho, Saiwai-ku, Kawasaki-shi,
Kanagawa 210 Japan
E-mail: {eriguchi, tkitani}@lit.rd.nttdata.jp
Phone: +81-44-548-4606

1 INTRODUCTION

Erie is a name recognition system developed for the Multilingual Entity Task (MET) in MUC-6. The pattern matching engine recognizes organization, person, and place names along with time and numeric expressions in Japanese text. Although our previous information extraction system Textract performed well in MUC-5, the pattern matching engine, which was written in AWK language, was slow[2]. System maintenance was also difficult, since the patterns were defined in both the matching engine and the pattern files. Erie solves these problems by generating a pattern matching engine in C language directly from the defined patterns.

2 SYSTEM DESCRIPTION

Figure 1 shows Erie's system architecture, which includes the following functions:

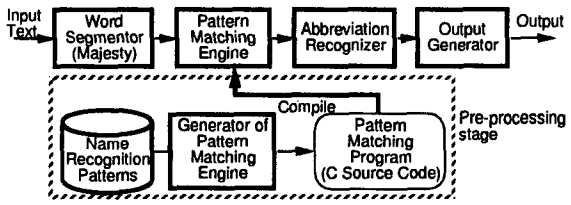

Figure 1 Erie system architecture

(1) Majesty is used to segment the Japanese text into primitive words and tags the parts of speech[1].
(2) Task-specific patterns are defined to modify the Majesty segmentation and to augment the parts of speech.
(3) Name recognition patterns can be defined in a form similar to regular expressions.
(4) An engine generator converts the defined patterns into a pattern matching program generated in C language.
(5) Abbreviations in the text are identified by an abbreviation recognizer.

3 PATTERNS

This section describes three types of patterns introduced in the previous section.

3.1 Dictionary patterns

Majesty tags a part of speech, such as a noun or noun-suffix, as the major category of the word. Then the dictionary pattern is used to add a sub-category to the word. The sub-category, for example that it is an organization, is defined on the left side of the pattern. Words to which the sub-category is added are listed on the right side of the pattern. Figure 2 shows an example of a dictionary pattern. The words "社" (a corporation) and "省" (a government ministry) are tagged as noun-suffixes (SUFFIX) by Majesty, while the dictionary pattern augments it by adding ORGANIZATION as its sub-category.

```
DICTIONARY {
  SUFFIX-ORGANIZATION = {社 | 省}
}
```

Figure 2 Example of a dictionary pattern

3.2 Segmentation patterns

The segmentation pattern is used to further segment a word whose word boundary is given by Majesty. The word to be segmented is written on the left side of the pattern. Newly-segmented words and their parts of speech are defined in the right side of the pattern. The pattern matching conditions of the matched word can be described in parenthesis. These conditions can be the part of speech of the word, the word preceding or following the word, or the word length. The character

'_' is a wild card that can match any number of characters within the word.

Figure 3 shows an example of the segmentation patterns. The first pattern divides a word "日米" (Japan and the U.S.) into "日" (Japan) and "米" (the U.S.), and gives each word a NOUN-PLACE tag as the part of speech. The second pattern divides a word whose last character is "相" (a government minister) into "相" and the rest of the word, if the word consists of more than three characters.

```
SEGMENTATION {
  日米 = 日:NOUN-PLACE 米:NOUN-PLACE
     {};
  _相 =  _:NOUN-ORGANIZATION
       相:SUFFIX-POSITION
       {LEN >= 3};
}
```

Figure 3 Example of segmentation patterns

3.3 Name recognition patterns

The name recognition patterns recognize proper names, times, and numeric expressions that appear in the text. A pattern name is written on the left side of the pattern, and the word sequence to be searched for is defined on the right side. The defined pattern can be referred to from other patterns by using the character '$' followed by the pattern name. A pattern can be any combination of words, their parts of speech, character type, and the pattern name. Regular expressions such as '*' and '+' can also be used in the pattern. Two angle brackets on the right side of the pattern specify the first and last of the words that comprise the identified name or expression. Figure 4 shows an example of a name recognition pattern that identifies a person's name.

```
PATTERN {
  $PERSON = < (NOUN | UNKNOWN)+ >
            SUFFIX-PERSON;
}
```

Figure 4 Example of a name recognition pattern

Erie's pattern matching engine processes the patterns in the order of definition. The first pattern that matches is chosen for the string currently being processed. Thus, pattern developers must pay special attention to the order of the patterns.

4 PATTERNS DEFINED IN ERIE

There are 54 dictionary patterns, 86 segmentation patterns, and 162 name recognition patterns defined in Erie. The pattern set was developed by using a hundred newspaper articles annotated and provided to the MET participants by DARPA.

During its official run on a Sun SparcStation 10, Erie processed each article in an average of 1.5 seconds. This is several times faster than Textract. But, entity names, especially person names, were not identified well, although time and numeric expressions were identified with a high level of recall and precision. This was probably because the patterns for entity names were not well enough defined. Since names can be expressed in many ways, a hundred newspaper articles used for the pattern development were insufficient.

5 OBSERVATIONS

Erie achieved a high processing accuracy in the Japanese MET task. In the course of this project, most of our time was spent on the development of the engine generator. Considering that the pattern development was done in only two weeks, our scores are quite satisfactory. This was achieved by separating the patterns and pattern matching engine, which has made the pattern development faster and easier. The pattern definition in Erie was powerful enough to identify the names and expressions required in the MET task.

The pattern development was mainly done by hand, which is very time-consuming. To develop systems more rapidly, tools are needed that will help pattern developers find and define patterns, then check the results. We will continue to work towards this goal and plan to improve our pattern matching engine to deal with more complicated patterns that Erie cannot currently handle.

References

[1] Kitani, T., Eriguchi, Y. and Hara, M. *"Pattern Matching and Discourse Processing Information Extraction from Japanese Text."* Journal of Artificial Intelligence Research, Vol. 2, pp. 89-110, 1994.

[2] Kitani, T. and Mitamura, T. *"An Accurate Morphological Analysis and Proper Name Identification for Japanese Text Processing."* Journal of Information Processing Society of Japan, 35(3), pp. 404-413, 1994.