Abstract

One of the central issues for information extraction (IE) systems is the cost of customization from one scenario to another. Research on the automated acquisition of patterns is important for portability and scalability. This paper explores the automatic extraction of patterns in Japanese from unannotated text. We introduce two modules of our system, the pattern extraction module and the information extraction module, both of which use structural patterns. The performance of the whole system is measured in a MUC-style evaluation.

1 Introduction

Information Extraction systems today are commonly based on pattern matching. New patterns need to be written when we customize an IE system for a new scenario (extraction task); this is costly if done by hand. This has led to recent research on automated acquisition of patterns from text with minimal pre-annotation. Riloff (Riloff, 1996) reported a successful result on her procedure that needs only a pre-classified corpus. Yangarber (Yangarber et al., 2000) proposed a procedure for unannotated natural language texts.

In general, Riloff and Yangarber relied on the sentence structure of English. Riloff predefined heuristic rules to create patterns based on syntactic structure, such as “<subj> active-verb” and “active-verb <dobj>”. Yangarber used triples of a predicate and some of its arguments, such as “<subj> <verb> <obj>”. However, there is a problem for a language with a flexible case marking system, like Japanese. Especially, we found that, in Japanese, some of the arguments that are usually marked as object in English were variously marked by different post-positions and the same case marker (postposition) marked more than one grammatical category in different situations. For example, the topic marker in Japanese, “wa”, can mark almost any entity that would have been variously marked in English. It is difficult to deal with this variety by simply fixing the number of arguments of a predicate for creating patterns in Japanese.

Moreover, Japanese use a lot of zero-pronouns, which makes the IE system fail to fill some of the slots unless these zero-pronouns are resolved. It is not uncommon that most predicates in Japanese sentences omit some of their arguments. For example, it is essential that a subject appears in English. However, in Japanese, it is not only grammatical but also usual to omit a subject of a sentence as long as it is understood from the context.

Another problem lies in relationships beyond clause boundaries, especially if the event is described in a subordinate clause. For example, for a sentence like “<organization> announced that <person> retired from <post>,” it is hard to find a relationship between <organization> and the event of retiring without the global view from the predicate “announce”.

In this paper, we propose using structural pattern matching to remedy the problems stated above. Unlike most pattern-based IE systems using patterns which rely on the surface word-
ordering, we used structural pattern matching where the system parses every sentence into a dependency tree and then does information extraction, as Figure 1 illustrates. This approach has two major advantages.

- **Deal with free word-ordering**
  The easiest way to overcome the problem of free word-ordering is to permute the base rule, so that the permutations can cover every possible word order in the language. However, this increases the number of generated patterns dramatically. Since the underlying dependency structure of a sentence is the same regardless the order of its words, one pattern of structural matching covers any possible word-ordering.

- **Capture structural relationships**
  The structural pattern can explicitly state the relationship between a predicate and its argument, for which the distance on the surface level is large. For example, in Figure 1, the structural pattern can clearly capture the direct dependency relationship from B to F, while the word-order pattern has to have a dummy element to match between B and F.

(Word-order Pattern)
Pattern [B * F]

![Pattern Variety]

(Structural Pattern)

![Pattern Variety]

2 Framework

2.1 Information Extraction Task
Information Extraction here is understood in the sense in MUC (Message Understanding Conference) literature. The task of IE is to extract the meaning of the documents in natural language.

Domain denotes the class of documents, and the scenario means the set of events to be extracted. Scenario Template task is one of the tasks in MUC, where the system is to identify the entities and their relationships in terms of events to fill the tabular form of template defined for each scenario.

In this paper, we set the scenario for our scenario template task, as “executive succession”, the topic of MUC-6 (DARPA, 1995), where the systems are to identify the events in which corporate managers left their positions and assumed the new positions. The entities here are defined as “person”, “organization”, “post” and the events have the property of “in and out” that determines if the person starts or leaves a position.

2.2 System Architecture

Our system consists of two major components, the pattern extraction module and information extraction module. Figure 2 shows the overall picture. Pattern extraction is the module that, given the training document set which includes documents relevant to the scenario, retrieves the relevant document set and the relevant sentence set and finds the patterns from the relevant sentence set. This will be discussed further in section 3. The information extraction module is to do structural pattern matching to the test data and fill the slots in the template, which we will discuss in section 4.

![System Architecture]

3 Pattern Extraction

In this section, we outline the procedure for the automatic pattern extraction module. First of all, the system does preprocessing on the whole
source document collection as is described in section 3.1. Then, given a description of the scenario of interest, the system retrieves a set of documents (section 3.2), a relevant document set, and a set of sentences (section 3.3), a relevant sentence set, in a cascading manner. Finally the system extracts a set of patterns from the relevant sentence set, as explained in section 3.4.

3.1 Document Preprocessing

Morphological analysis and Named Entity (NE) tagging is performed on the training data at this stage. All the words in the document collection is segmented by the Japanese Morphological Analyzer, JUMAN (JUMAN, 1998). POS information is also given by JUMAN and is used in the later stages. We ran a NE-system which is based on a decision tree algorithm (Sekine et al., 1998) on the document collection to tag all the expressions of person names, organization names, locations, dates, money, percent, and post.

Having all the documents in the collection, the system calculates the document frequency of every word and NE.

3.2 Document Retrieval

The system first retrieves the documents that describe the events of the scenario of interest, called the relevant document set. A set of narrative sentences describing the scenario is selected to create a query for the retrieval. For this experiment, we retrieved the documents by CRL’s stochastic-model-based IR system (Murata et al., 1999), which performed well in the IR task in IREX, Information Retrieval and Extraction evaluation project in Japan. All the sentences used to create the patterns are retrieved from this relevant document set.

3.3 Sentence Retrieval

The system then calculates the TF/IDF-based score of relevance to the scenario for each sentence in the relevant document set and retrieves the n most relevant sentences as the source of the patterns, where n is set to 1500 for this experiment. The retrieved sentences will be the source for pattern extraction in the next subsection.

First, the TF/IDF-based score for every word in the relevant document set is calculated. TF/IDF score of word w is:

\[ \text{score}(w) = \begin{cases} 
T F(w) \cdot \frac{\log(N + 0.5)}{\log(1 + T F(w))} & \text{if } w \text{ is Noun, Verb or Named Entity} \\
0 & \text{otherwise}
\end{cases} \]

where N is the number of documents in the collection, TF(w) is the term frequency of w in the relevant document set and DF(w) is the document frequency of w in the collection.

Second, the system calculates the score of each sentence based on the score of its words. However, unusually short sentences and unusually long sentences will be penalized. TF/IDF score of sentence s is:

\[ \text{score}(s) = \frac{\sum_{w \in s} \text{score}(w)}{\text{length}(s) + \left| \text{length}(s) - \text{AVE} \right|} \]

where length(s) is the number of words in s, and AVE is the average number of words in a sentence.

3.4 Pattern Construction

Based on the dependency trees of the sentences, patterns are extracted from the relevant sentences retrieved in the previous subsection. Figure 3 shows the procedure. First, the retrieved sentence is parsed into a dependency tree by KNP (Kurohashi and Nagao, 1994) (Stage 1). It also finds predicates in the tree. Second, the system takes all the predicates in the tree as the roots of their own subtrees, as is shown in (Stage 2). Then each path from the root to a node is counted separately. These counts are accumulated across all relevant sentences, and the paths are sorted in order of frequency. Finally, the system takes those paths with frequency higher than some threshold as extracted patterns.

3.5 Additional Information

For this specific task “executive succession”, we also need to fill in and out slots (whether the person comes in to a position or goes out from the position). The current version of the system does not support for associating each pattern to in and out slot. Therefore, we need to tag the in and out attribute by hand on each pattern.

---

1IREX Homepage: http://cs.nyu.edu/cs/projects/proteus/irex/
4 Information Extraction

This section describes the extraction module we used for the evaluation of the patterns.

4.1 Preprocessing on the Test Data

As is done for pattern extraction, all the documents in the test corpus are preprocessed by the NE-tagger, the morphological analyzer, and the dependency analyzer, so that the system would get the dependency tree for each sentence in the document. Also, as the sentences are parsed, the hierarchy of the predicate nodes in the dependency tree (reduced tree with only the predicate nodes) is stored for later use in section 4.3.

4.2 Structural Pattern Matching

The pattern matching of this system is based on the dependency tree of each sentence. After finding all the predicates in a sentence, the system looks for the match of the path from each predicate node to its descendants of the tree. All the matched patterns are kept regardless their redundancy for the later stages. The system takes the expressions as slot-fillers of the template if they are in the leaf nodes and tagged as person name, organization name or post.

4.3 Template Merging

Since most of the patterns extracted in the previous section have one slot-filler and patterns can match various parts of the sentence, it is necessary to merge the matched patterns to fill the whole event template.

After matching each pattern separately, the system groups all the fillers of the template according to their predicates to make predicate node structures for each predicate. Figure 4 illustrates the example of the predicate node structure for the sentence “Sony (org)-wa Idei Nobuyoshi (person)-ga shacho (post)-ni shuninsuru-to happyoshita (Sony announced that Nobuyoshi Idei was appointed as chair).” 2

According to the predicate hierarchy of the original sentence, the system traverses the dependency hierarchy from the root predicate of the sentence. The predicate node inherits all the unassigned slots from its parent predicate node, if the slots in the parent predicate node do not have any conflict in in and out attribute with the unassigned slots of the child predicate node. 3

4.4 Coreference Resolution

The pattern merging does not resolve reference beyond sentence boundaries. Because of the presence of zero pronouns in Japanese, it is important that the reference of the zero pronouns should be

---

2In this figure, the pattern “<person>-ga shuninsuru (to be appointed)” is also marked as out because it is considered to indicate that the person left out from the previous position.

3In Figure 4, none is considered to conflict none of in and out attribute values.
resolved by the use of context. A simple approach is taken: the system keeps a list of the candidate fillers for each slot. If the matched pattern does not have an entity for a particular slot even after pattern merging, the system automatically takes the most recently referenced entity of the same type in the reference list.

5 Experiments

As we proposed above, the evaluation was done by applying the automatically acquired patterns using our IE-engine. First, pattern extraction was performed using the training data. Second, information extraction with the acquired patterns was done on the test data. The evaluation is based on the MUC-6 scenario-template (ST) task. We simplified the task by using only a few core slots in the template, namely: person name, organization name, post, new status (in or out), succession event. The result is compared to the performance with manually created patterns on the same data (Nobata and Sekine, 1999), which they used as the training data of their system.

5.1 Data

Nikkei Newspaper for 1995 was used for the pattern extraction module and 87 articles from Nikkei Newspaper 1994 were used as test data for the information extraction module. The system retrieved 1500 articles as the relevant document set and then retrieved 1500 sentences as the relevant sentence set. All the patterns were extracted from the relevant document set. We took 1089 paths that appeared more than once in the dependency trees of the relevant sentence set as extracted patterns.

5.2 Results

The result of this experiment was analyzed quantitatively. We evaluated the performance by precision and recall in terms of the slot-fillers in MUC literature. To get different points on the precision-recall curve, we ranked all the extracted patterns by calculating the sum of the TF/IDF-based score (same as for sentence retrieval in Section 3.3) for each word in the pattern and sorting them on this basis, and calculate the precision and recall for different numbers of top relevant patterns.

Figure 5 shows the precision-recall curve of this experiment. The result was calculated by using the MUC-6 scorer configured to use only the core slots mentioned above. The curve was obtained by changing the number of top-ranked patterns used for extraction.

When compared to the high recall for the manual patterns (73%), our system achieved low recall (42%) which will be discussed in the next section. The system got around 65% precision while the manual system’s precision is 72%. Note, however, that the manual system’s performance is on their training data and their separate test data was not available for this experiment.

6 Discussion

- Low Recall

We should note that the recall of our system is still low although the number of the patterns is large. It is especially difficult to capture the organization names, so that
Figure 5: Precision-Recall Curve

our system could extract only 15% of the organization-name slots. One of the reasons is that many of the organization-names are found in the test sentence where the path is too rare to extract as a pattern because of its depth. Among the cases where our system failed to fill the organization-name slots, the most descriptive example is the following sentence. “Apple Computer-de kenkyukaihatsu bumon-o toukatsushiteita David Nagel jookyuu-fukushachoo-ga taisha (Senior vice president, David Nagel, who administered the R&D section at Apple Computer Inc., retired.)” Here toukatsusuru (administer) did not appear enough often to extract the pattern “<org>-de → toukatsusuru → taininsuru (retire)”.

- **Necessity of Lexical Generalization**
We have not yet attempted any lexical generalization of pattern candidates. The patterns can be expanded by using a thesaurus and/or introducing a new lexical class suitable for a particular domain. Especially the generalized patterns will help improve recall.

7 Future Work

For the short term, we are working on customizing and porting this framework into a different domain and language. Similar framework for information extraction module may work for another domain, as well.

Our long term goal is to develop a Cross-Language Information Extraction system of English and Japanese. Especially the pattern extraction module proposed in this paper will be used as the monolingual IE module which is easy to customize for different languages since ax morphological analyzer, parser (dependency analyzer) and NE-tagger are the only components depending on the target language.

**Acknowledgements**

This research is supported by the Defense Advanced Research Projects Agency as part of the Translingual Information Detection, Extraction and Summarization (TIDES) program, under Grant N66001-00-1-8917 from the Space and Naval Warfare Systems Center San Diego and by the National Science Foundation under Grant IIS-0081962. This paper does not necessarily reflect the position or the policy of the U.S. Government.

**References**

DARPA. 1995. *Proceedings of the Sixth Message Understanding Conference (MUC-6).*


JUMAN homepage : http://www-nagao.kuee.kyoto-u.ac.jp/nl-resource/juman-e.html


Ellen Riloff. 1996. “ Automatically Generating Extraction Patterns from Untagged Text.” *In the Proceedings of Thirteenth National Conference on Artificial Intelligence (AAAI-96).*
