Learning Semantic-Level Information Extraction Rules by Type-Oriented ILP

Yutaka Sasaki and **Yoshihiro Matsuo**
NTT Communication Science Laboratories
2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237, Japan
{sasaki, yosihiro}@cslab.kecl.ntt.co.jp

Abstract

This paper describes an approach to using *semantic representations* for learning *information extraction (IE)* rules by a type-oriented *inductive logic programming (ILP)* system. NLP components of a machine translation system are used to automatically generate semantic representations of text corpus that can be given directly to an ILP system. The latest experimental results show high precision and recall of the learned rules.

1 Introduction

Information extraction (IE) tasks in this paper involve the MUC-3 style IE. The input for the information extraction task is an empty *template* and a set of natural language texts that describe a restricted target domain, such as corporate mergers or terrorist attacks in South America. Templates have a record-like data structure with slots that have names, *e.g.*, "company name" and "merger date", and values. The output is a set of filled templates. IE tasks are highly domain-dependent, so rules and dictionaries for filling values in the template slots depend on the domain.

It is a heavy burden for IE system developers that such systems depend on hand-made rules, which cannot be easily constructed and changed. For example, Umass/MUC-3 needed about 1,500 person-hours of highly skilled labor to build the IE rules and represent them as a dictionary (Lehnert, 1992). All the rules must be reconstructed from scratch when the target domain is changed.

To cope with this problem, some pioneers have studied methods for learning information extraction rules (Riloff,1996; Soderland *et al.*, 1995; Kim *et al.*, 1995; Huffman, 1996; Califf and Mooney, 1997). Along these lines, our approach is to apply an *inductive logic programming (ILP)* (Muggleton, 1991) system to the learning of IE rules, where information is extracted from semantic representations of news articles. The ILP system that we employed is a type-oriented ILP system RHB$^+$ (Sasaki and Haruno, 1997), which can efficiently and effectively handle type (or sort) information in training data.

2 Our Approach to IE Tasks

This section describes our approach to IE tasks. Figure 1 is an overview of our approach to learning IE rules using an ILP system from semantic representations. First, training articles are analyzed and converted into *semantic representations*, which are filled *case frames* represented as atomic formulae. Training templates are prepared by hand as well. The ILP system learns IE rules in the form of logic programs with type information. To extract key information from a new article, semantic representations automatically generated from the article is matched by the IE rules. Extracted information is filled into the template slots.

3 NLP Resources and Tools

3.1 The Semantic Attribute System

We used the semantic attribute system of "Goi-Taikei — A Japanese Lexicon" (Ikehara *et al.*, 1997a; Kurohashi and Sakai, 1999) compiled by the NTT Communication Science Laboratories for a Japanese-to-English machine translation system, ALT-J/E (Ikehara *et al.*, 1994). The semantic attribute system is a sort of hierarchical concept thesaurus represented as a tree structure in which each node is called a *semantic category*. An edge in the tree represents an *is_a* or *has_a* relation between two categories. The semantic attribute system is 12 levels deep and

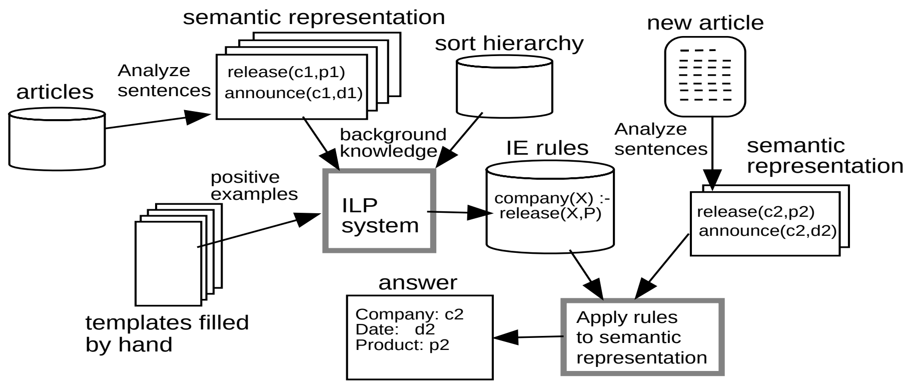

Figure 1: Block diagram of IE using ILP

contains about 3,000 semantic category nodes. More than 300,000 Japanese words are linked to the category nodes.

3.2 Verb Case Frame Dictionary

The Japanese-to-English valency pattern dictionary of "Goi-Taikei" (Ikehara *et al.*, 1997b; Kurohashi and Sakai, 1999) was also originally developed for ALT-J/E. The valency dictionary contains about 15,000 case frames with semantic restrictions on their arguments for 6,000 Japanese verbs. Each case frame consists of one predicate and one or more case elements that have a list of semantic categories.

3.3 Natural Language Processing Tools

We used the NLP components of ALT-J/E for text analysis. These include the morphological analyzer, the syntactic analyzer, and the case analyzer for Japanese. The components are robust and generic tools, mainly targeted to newspaper articles.

3.3.1 Generic Case Analyzer

Let us examine the case analysis in more detail. The case analyzer reads a set of parse tree candidates produced by the Japanese syntactic analyzer. The parse tree is represented as a dependency of phrases (*i.e.*, Japanese *bunsetsu*).

First, it divides the parse tree into unit sentences, where a unit sentence consists of one predicate and its noun and adverb dependent phrases. Second, it compares each unit sentence with a verb case frame dictionary. Each frame consists a predicate condition and several case elements conditions. The predicate condition specifies a verb that matches the frame and each case-role has a case element condition which specifies particles and semantic categories of noun phrases. The preference value is defined as the summation of noun phrase preferences which are calculated from the distances between the categories of the input sentences and the categories written in the frames. The case analyzer then chooses the most preferable parse tree and the most preferable combination of case frames.

The valency dictionary also has case-roles (Table 1) for noun phrase conditions. The case-roles of adjuncts are determined by using the particles of adjuncts and the semantic categories of noun phrases.

As a result, the output of the case analysis is a set of case frames for each unit sentence. The noun phrases in frames are labeled by case-roles in Table 1.

For simplicity, we use case-role codes, such as N1 and N2, as the labels (or slot names) to represent case frames. The relation between sentences and case-roles is described in detail in (Ikehara *et al.*, 1993).

3.3.2 Logical Form Translator

We developed a logical form translator FEP that generates semantic representations expressed as atomic formulae from the case frames and parse trees. For later use, document ID and tense information are also added to the case frames.

For example, the case frame in Table 2 is obtained after analyzing the following sentence of document D1:

"*Jakku*(Jack) *ha* *sutsukesu*(suitcase) *wo*

Table 1: Case-Roles

Name	Code	Description	Example
Subject	N1	the agent/experiencer of an event/situation	*I* throw a ball.
Object1	N2	the object of an event	I throw *a ball.*
Object2	N3	another object of an event	I compare it *with them.*
Loc-Source	N4	source location of a movement	I start *from Japan.*
Loc-Goal	N5	goal location of a movement	I go *to Japan.*
Purpose	N6	the purpose of an action	I go *shopping.*
Result	N7	the result of an event	It results *in failure.*
Locative	N8	the location of an event	It occurs *at the station.*
Comitative	N9	co-experiencer	I share a room *with him.*
Quotative	N10	quoted expression	I say *that ....*
Material	N11	material/ingredient	I fill the glass *with water.*
Cause	N12	the reason for an event	It collapsed *from the weight.*
Instrument	N13	a concrete instrument	I speak *with a microphone.*
Means	N14	an abstract instrument	I speak *in Japanese.*
Time-Position	TN1	the time of an event	I go to bed *at 10:00.*
Time-Source	TN2	the starting time of an event	I work *from Monday.*
Time-Goal	TN3	the end time of an event	It continues *until Monday.*
Amount	QUANT	quantity of something	I spend *$10.*

shokuba(the office) *kara*(from) *kuko*(the airport) *ni*(to) *hakobu*(carry)"

("Jack carries a suitcase from the office to the airport.")

Table 2: Case Frame of the Sample Sentence

predicate:	*hakobu* (carry)
article:	D1
tense:	present
N1:	*Jakku* (Jack)
N2:	*sutsukesu* (suitcase)
N4:	*shokuba* (the office)
N5:	*kuko* (the airport)

4 Inductive Learning Tool

Conventional ILP systems take a set of positive and negative examples, and background knowledge. The output is a set of hypotheses in the form of logic programs that covers positives and do not cover negatives. We employed the type-oriented ILP system RHB$^+$.

4.1 Features of Type-oriented ILP System RHB$^+$

The type-oriented ILP system has the following features that match the needs for learning IE rules.

- A type-oriented ILP system can efficiently and effectively handle type (or semantic category) information in training data. This feature is advantageous in controlling the generality and accuracy of learned IE rules.
- It can directly use semantic representations of the text as background knowledge.
- It can learn from only positive examples.
- Predicates are allowed to have labels (or keywords) for readability and expressibility.

4.2 Summary of Type-oriented ILP System RHB$^+$

This section summarizes the employed type-oriented ILP system RHB$^+$. The input of RHB$^+$ is a set of positive examples and background knowledge including type hierarchy (or the semantic attribute system). The output is a set of Horn clauses (Lloyd, 1987) having variables with type information. That is, the term is extended to the τ-term.

4.3 τ-terms

τ-terms are the restricted form of ψ-terms (Aït-Kaci and Nasr, 1986; Aït-Kaci *et al.*, 1994). Informally, τ-terms are Prolog terms whose variables are replaced with variable *Var* of type T, which is denoted as *Var:T*. Predicate and function symbols are allowed to have features (or labels). For example,

$speak(agent \Rightarrow X{:}human, object \Rightarrow Y{:}language)$
is a clause based on τ-terms which has labels *agent* and *object*, and types *human* and *language*.

4.4 Algorithm

The algorithm of RHB$^+$ is basically a greedy covering algorithm. It constructs clauses one-by-one by calling inner_loop (Algorithm 1) which returns a hypothesis clause. A hypothesis clause is represented in the form of *head* :− *body*. Covered examples are removed from P in each cycle.

The inner loop consists of two phases: the head construction phase and the body construction phase. It constructs heads in a bottom-up manner and constructs the body in a top-down manner, following the result described in (Zelle *et al.*, 1994).

The search heuristic *PWI* is weighted informativity employing the Laplace estimate. Let $T = \{Head :- Body\} \cup BK$.

$$PWI(P,T) = -\frac{1}{|\hat{P}|} \times \log_2 \frac{|\hat{P}|+1}{|Q(T)|+2},$$

where $|\hat{P}|$ denotes the number of positive examples covered by T and $Q(T)$ is the empirical content. The smaller the value of PWI, the candidate clause is better. $Q(T)$ is defined as the set of atoms (1) that are derivable from T and (2) whose predicate is the target predicate, *i.e.*, the predicate name of the head.

The *dynamic type restriction by positive examples* uses positive examples currently covered in order to determine appropriate types to variables for the current clause.

Algorithm 1 *inner_loop*

```
1. Given positives P, original positives $P_0$, background knowledge BK.
2. Decide types of variables in a head by computing the typed least general generalizations (lgg) of N pairs of elements in P, and select the most general head as Head.
3. If the stopping condition is satisfied, return Head.
4. Let Body be empty.
5. Create a set of all possible literals L using variables in Head and Body.
6. Let BEAM be top K literals $l_k$ of L with respect to the positive weighted informativity PWI.
7. Do later steps, assuming that $l_k$ is added to Body for each literal $l_k$ in BEAM.
8. Dynamically restrict types in Body by calling the dynamic type restriction by positive examples.
9. If the stopping condition is satisfied, return (Head :− Body).
10. Goto 5.
```

5 Illustration of a Learning Process

Now, we examine the two short notices of new products release in Table 3. The following table shows a sample template for articles reporting a new product release.

Template
1. article id:
2. company:
3. product:
4. release date:

5.1 Preparation

Suppose that the following semantic representations is obtained from Article 1.

```
(c1) announce( article => 1,
       tense => past,
       tn1  => "this week",
       n1    => "ABC Corp.",
       n10   => (c2) ).

(c2) release( article => 1,
         tense => future,
        tn1   => "Jan. 20",
        n1     => "ABC Corp.",
        n2     => "a color printer" ).
```

Table 3: Sample Sentences

Article id	Sentence
#1	"ABC Corp. this week announced that it will release a color printer on Jan. 20."
#2	"XYZ Corp. released a color scanner last month."

The filled template for Article 1 is as follows.

Template 1
1. article id: 1
2. company: ABC Corp.
3. product: a color printer
4. release date: Jan. 20

Suppose that the following semantic representation is obtained from Article 2.

```
(c3) release( article => 2,
       tense => past,
       tn1  => "last month",
       n1    => "XYZ Corp.",
       n2    => "a color scanner" ).
```

The filled template for Article 2 is as follows.

Template 2
1. article id: 2
2. company: XYZ Corp.
3. product: a color scanner
4. release date: last month

5.2 Head Construction

Two positive examples are selected for the template slot "company".

```
company(article-number => 1
            name => "ABC Corp").

company(article-number => 2
            name => "XYZ Corp").
```

By computing a least general generalization (lgg)sasaki97, the following head is obtained:

```
company( article-number => Art: number
            name => Co: organization).
```

5.3 Body Construction

Generate possible literals[1] by combining predicate names and variables, then check the PWI values of clauses to which one of the literal added. In this case, suppose that adding the following literal with predicate *release* is the best one. After the dynamic type restriction, the current clause satisfies the stopping condition. Finally, the rule for extracting "company name" is returned. Extraction rules for other slots "product" and "release date" can be learned in the same manner. Note that several literals may be needed in the body of the clause to satisfy the stopping condition.

```
company(article-number => Art:number,
     name => Co: organization   )

   :- release( article => Art,
         tense => T: tense,
         tn1  => D: time,
         n1    => Co,
         n2    => P: product    ).
```

5.4 Extraction

Now, we have the following semantic representation extracted from the new article:
Article 3: "JPN Corp. has released a new CD player."[2]

```
(c4) release( article => 3,
        tense => perfect_present,
        tn1  => nil,
        n1    => "JPN Corp.",
        n2    => "a new CD player" ).
```

Applying the learned IE rules and other rules, we can obtain the filled template for Article 3.

Template 3
1. article id: 3
2. company: JPN Corp.
3. product: CD player
4. release date:

[1]"literals" here means atomic formulae or negated ones.

[2]We always assume *nil* for the case that is not included in the sentence.

Table 4: Learning results of new product release

(a) Without data correction

	company	product	release date	announce date	price
Precision	89.6%	80.5%	90.6%	100.0%	58.4%
Recall	82.1%	66.7%	66.7%	82.4%	60.8%
Average time (sec.)	15.8	22.1	14.4	2.2	11.0

(b) With data correction

	company	product	release date	announce date	price
Precision	91.1%	80.0%	92.3%	100.0%	87.1%
Recall	85.7%	69.7%	82.8%	88.2%	82.4%
Average time (sec.)	22.9	25.2	33.55	5.15	11.9

6 Experimental Results

6.1 Setting of Experiments

We extracted articles related to the release of new products from a one-year newspaper corpus written in Japanese [3]. One-hundred articles were randomly selected from 362 relevant articles. The template we used consisted of five slots: *company name*, *product name*, *release date*, *announce date*, and *price*. We also filled one template for each article. After analyzing sentences, case frames were converted into atomic formulae representing semantic representations as described in Section 2 and 3. All the semantic representations were given to the learner as background knowledge, and the filled templates were given as positive examples. To speed-up the learning process, we selected predicate names that are relevant to the words in the templates as the target predicates to be used by the ILP system, and we also restricted the number of literals in the body of hypotheses to one.

Precision and recall, the standard metrics for IE tasks, are counted by using the remove-one-out cross validation on the examples for each item. We used a VArStation with the Pentium II Xeon (450 MHz) for this experiment.

6.2 Results

Table 4 shows the results of our experiment. In the experiment of learning from semantic representations, including errors in case-role selection and semantic category selection, precision was very high. The precision of the learned rules for price was low because the semantic category name automatically given to the price expressions in the data were not quite appropriate. For the five items, 67-82% recall was achieved. With the background knowledge having semantic representations corrected by hand, precision was very high and 70-88% recall was achieved. The precision of price was markedly improved.

It is important that the extraction of five different pieces of information showed good results. This indicates that the ILP system RHB$^+$ has a high potential in IE tasks.

[3]We used articles from the Mainichi Newspapers of 1994 with permission.

7 Related Work

Previous researches on generating IE rules from texts with templates include AutoSlog-TS (Riloff,1996), CRYSTAL (Soderland *et al.*, 1995), PALKA (Kim *et al.*, 1995), LIEP (Huffman, 1996) and RAPIER (Califf and Mooney, 1997). In our approach, we use the type-oriented ILP system RHB$^+$, which is independent of natural language analysis. This point differentiates our approach from the others. Learning semantic-level IE rules using an ILP system from semantic representations is also a new challenge in IE studies.

Sasaki (Sasaki and Haruno, 1997) applied RHB$^+$ to the extraction of the number of deaths and injuries from twenty five articles. That experiment was sufficient to assess the performance of the learner, but not to evaluate its feasibility in IE tasks.

8 Conclusions and Remarks

This paper described a use of semantic representations for generating information extraction rules by applying a type-oriented ILP system. Experiments were conducted on the data generated from 100 news articles in the domain of new product release. The results showed very high precision, recall of 67-82% without data correction and 70-88% recall with correct semantic representations. The extraction of five different pieces of information showed good results. This indicates that our learner RHB$^+$ has a high potential in IE tasks.

References

H. Aït-Kaci and R. Nasr, LOGIN: A logic programming language with built-in inheritance, *Journal of Logic Programming*, 3, pp.185–215, 1986.

H. Aït-Kaci, B. Dumant, R. Meyer, A. Podelski, and P. Van Roy, *The Wild Life Handbook*, 1994.

M. E. Califf and R. J. Mooney, Relational Learning of Pattern-Match Rules for Information Extraction, *Proc. of ACL-97 Workshop in Natural Language Learning*, 1997.

S. B. Huffman, Learning Information Extraction Patterns from Examples, *Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pp.246–260, 1996.

S. Ikehara, M. Miyazaki, and A. Yokoo, Classification of language knowledge for meaning analysis in machine translations, *Transactions of Information Processing Society of Japan*, Vol.34, pp.1692-1704, 1993. (in Japanese)

S. Ikehara, S. Shirai, K. Ogura, A. Yokoo, H. Nakaiwa and T. Kawaoka, ALT-J/E: A Japanese to English Machine Translation System for Communication with Translation, *Proc. of The 13th IFIP World Computer Congress*, pp.80–85, 1994.

S. Ikehara, M. Miyazaki, S. Shirai, A. Yokoo, H. Nakaiwa, K. Ogura, Y. Oyama and Y. Hayashi (eds.), The Semantic Attribute System, *Goi-Taikei — A Japanese Lexicon*, Vol.1, Iwanami Publishing, 1997. (in Japanese)

S. Ikehara, M. Miyazaki, S. Shirai, A. Yokoo, H. Nakaiwa, K. Ogura, Y. Oyama and Y. Hayashi (eds.), The Valency Dictionary, *Goi-Taikei — A Japanese Lexicon*, Vol.5, Iwanami Publishing, 1997. (in Japanese)

J.-T. Kim and D. I. Moldovan, Acquisition of Linguistic Patterns for Knowledge-Based Information Extraction, *IEEE Transaction on Knowledge and Data Engineering (IEEE TKDE)*, Vol.7, No.5, pp.713–724, 1995.

S. Kurohashi and Y. Sakai, Semantic Analysis of Japanese Noun Phrases: A New Approach to Dictionary-Based Understanding *The 37th Annual Meeting of the Association for Computational Linguistics (ACL-99)*, pp.481–488, 1999.

W. Lehnert, C. Cardie, D. Fisher, J. McCarthy, E. Riloff and S. Soderland, University of Massachusetts: MUC-4 Test Results and Analysis, *Proc. of The Fourth Message Understanding Conference (MUC-4)*, pp.151–158, 1992.

J. Lloyd, *Foundations of Logic Programming*, Springer, 1987.

S. Muggleton, Inductive logic programming, *New Generation Computing*, Vol.8, No.4, pp.295-318, 1991.

E. Riloff, Automatically Generating Extraction Pattern from Untagged Text, *Proc.of American Association for Artificial Intelligence (AAAI-96)*, pp.1044–1049, 1996.

Y. Sasaki and M. Haruno, RHB$^+$: A Type-Oriented ILP System Learning from Positive Data, *Proc. of The 14th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pp.894–899, 1997.

S. Soderland, D. Fisher, J. Aseltine, W. Lenert, CRYSTAL: Inducing a Conceptual Dictionary, *Proc. of The 13th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pp.1314–1319, 1995.

J. M. Zelle and R. J. Mooney, J. B. Konvisser, Combining Top-down and Bottom-up Methods in Inductive Logic Programming, *Proc of The 11th International Conference on Machine Learning (ML-94)*, pp.343–351, 1994.