An Improved Extraction Pattern Representation Model for Automatic IE Pattern Acquisition

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman
Department of Computer Science
New York University
715 Broadway, 7th Floor,
New York, NY 10003 USA
{sudo,sekine,grishman}@cs.nyu.edu

Abstract

Several approaches have been described for the automatic unsupervised acquisition of patterns for information extraction. Each approach is based on a particular model for the patterns to be acquired, such as a predicate-argument structure or a dependency chain. The effect of these alternative models has not been previously studied. In this paper, we compare the prior models and introduce a new model, the Subtree model, based on arbitrary subtrees of dependency trees. We describe a discovery procedure for this model and demonstrate experimentally an improvement in recall using Subtree patterns.

1 Introduction

Information Extraction (IE) is the process of identifying events or actions of interest and their participating entities from a text. As the field of IE has developed, the focus of study has moved towards automatic knowledge acquisition for information extraction, including domain-specific lexicons (Riloff, 1993; Riloff and Jones, 1999) and extraction patterns (Riloff, 1996; Yangarber et al., 2000; Sudo et al., 2001). In particular, methods have recently emerged for the acquisition of event extraction patterns without corpus annotation in view of the cost of manual labor for annotation. However, there has been little study of alternative representation models of extraction patterns for unsupervised acquisition.

In the prior work on extraction pattern acquisition, the representation model of the patterns was based on a fixed set of pattern templates (Riloff, 1996), or predicate-argument relations, such as subject-verb, and object-verb (Yangarber et al., 2000). The model of our previous work (Sudo et al., 2001) was based on the paths from predicate nodes in dependency trees.

In this paper, we discuss the limitations of prior extraction pattern representation models in relation to their ability to capture the participating entities in scenarios. We present an alternative model based on subtrees of dependency trees, so as to extract entities beyond direct predicate-argument relations. An evaluation on scenario-template tasks shows that the proposed Subtree model outperforms the previous models.

Section 2 describes the Subtree model for extraction pattern representation. Section 3 shows the method for automatic acquisition. Section 4 gives the experimental results of the comparison to other methods and Section 5 presents an analysis of these results. Finally, Section 6 provides some concluding remarks and perspective on future research.

2 Subtree model

Our research on improved representation models for extraction patterns is motivated by the limitations of the prior extraction pattern representations. In this section, we review two of the previous models in detail, namely the Predicate-Argument model (Yangarber et al., 2000) and the Chain model (Sudo et al., 2001).

The main cause of difficulty in finding entities by extraction patterns is the fact that the participating entities can appear not only as an argument of the predicate that describes the event type, but also in other places within the sentence or in the prior text. In the MUC-3 terrorism scenario, WEAPON entities occur in many different relations to event predicates in the documents. Even if WEAPON entities appear in the same sentence with the event predicate, they rarely serve as a direct argument of such predicates. (e.g., "One person was *killed* as the result of a *bomb* explosion.")

Predicate-Argument model The Predicate-Argument model is based on a direct syntactic relation between a predicate and its arguments[1] (Yangarber et al., 2000). In general, a predicate provides a strong context for its arguments, which leads to good accuracy. However, this model has two major limitations in terms of its coverage, clausal boundaries and embedded entities inside a predicate's arguments.

Figure 1[2] shows an example of an extraction task in the terrorism domain where the event template consists of *perpetrator*, *date*, *location* and *victim*. With the extraction patterns based on the Predicate-Argument model, only *perpetrator* and *victim* can be extracted. The *location* (*downtown Jerusalem*) is embedded as a modifier of the noun (*heart*) within the prepositional phrase, which is an adjunct of the main predicate, *triggered*[3]. Furthermore, it is not clear whether the extracted entities are related to the same event, because of the clausal boundaries.[4]

[1]Since the case marking for a nominalized predicate is significantly different from the verbal predicate, which makes it hard to regularize the nominalized predicates automatically, the constraint for the Predicate-Argument model requires the root node to be a *verbal* predicate.

[2]Throughout this paper, extraction patterns are defined as one or more word classes with their context in the dependency tree, where the actual word matched with the class is associated to one of the slots in the template. The notation of the patterns in this paper is based on a dependency tree where (x (y_1-t_1)..(y_n-t_n)) denotes x is the head, and, for each i in $1..n$, y_i is its argument and the relation between x and y_i is labeled with t_i. The labels introduced in this paper are SBJ (subject), OBJ (object), ADV (adverbial adjunct), REL (relative), APPOS (apposition) and prepositions (IN, OF, etc.). Also, we assume that the order of the arguments does not matter. Symbols beginning with C- represent NE (Named Entity) types.

[3]Yangarber refers this as a *noun phrase pattern* in (Yangarber et al., 2000).

[4]This is the problem of merging the result of entity extraction. Most IE systems have hard-coded inference rules, such as "*triggering an explosion* is related to *killing* or *injuring* and therefore constitutes one terrorism action."

Chain model Our previous work, the Chain model (Sudo et al., 2001)[5] attempts to remedy the limitations of the Predicate-Argument model. The extraction patterns generated by the Chain model are any chain-shaped paths in the dependency tree.[6] Thus it successfully avoids the clausal boundary and embedded entity limitation. We reported a 5% gain in recall at the same precision level in the MUC-6 management succession task compared to the Predicate-Argument model.

However, the Chain model also has its own weakness in terms of accuracy due to the lack of context. For example, in Figure 1(c), (triggered ({C-DATE}-ADV)) is needed to extract the *date* entity. However, the same pattern is likely to be applied to texts in other domains as well, such as "The Mexican peso was devalued and *triggered* a national financial crisis *last week*."

Subtree model The Subtree model is a generalization of previous models, such that any subtree of a dependency tree in the source sentence can be regarded as an extraction pattern candidate. As shown in Figure 1(d), the Subtree model, by its definition, contains all the patterns permitted by either the Predicate-Argument model or the Chain model. It is also capable of providing more relevant context, such as (triggered (explosion-OBJ)({C-DATE}-ADV)).

The obvious advantage of the Subtree model is the flexibility it affords in creating suitable patterns, spanning multiple levels and multiple branches. Pattern coverage is further improved by relaxing the constraint that the root of the pattern tree be a predicate node. However, this flexibility can also be a disadvantage, since it means that a very large number of pattern candidates — all possible subtrees of the dependency tree of each sentence in the corpus — must be considered. An efficient procedure is required to select the appropriate patterns from among the candidates.

Also, as the number of pattern candidates increases, the amount of noise and complexity in-

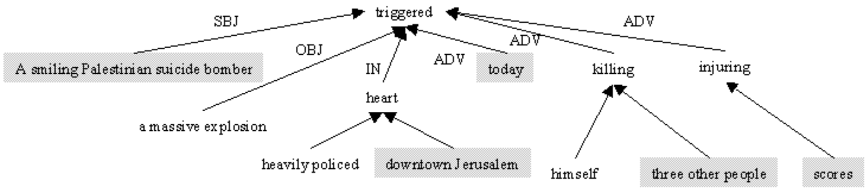
[5]Originally we called it "Tree-Based Representation of Patterns". We renamed it to avoid confusion with the proposed approach that is also based on dependency trees.

[6](Sudo et al., 2001) required the root node of the chain to be a *verbal* predicate, but we have relaxed that constraint for our experiments.

(a)

JERUSALEM, March 21 – A smiling Palestinian suicide bomber triggered a massive explosion in the heavily policed heart of downtown Jerusalem today, killing himself and three other people and injuring scores.

(b)



(c)

Predicate-Argument	Chain model
(triggered ({C-PERSON}-SBJ)(explosion-OBJ)({C-DATE}-ADV))	(triggered ({C-PERSON}-SBJ))
(killing ({C-PERSON}-OBJ))	(triggered (heart-IN ({C-LOCATION}-OF)))
(injuring ({C-PERSON}-OBJ))	(triggered (killing-ADV ({C-PERSON}-OBJ)))
	(triggered (injuring-ADV ({C-PERSON}-OBJ)))
	(triggered ({C-DATE}-ADV))

(d)

Subtree model	
(triggered ({C-PERSON}-SBJ)(explosion-OBJ))	(triggered (explosion-OBJ)({C-DATE}-ADV))
(killing ({C-PERSON}-OBJ))	(triggered ({C-DATE}-ADV)(killing-ADV))
(injuring ({C-PERSON}-OBJ))	(triggered ({C-DATE}-ADV)(killing-ADV({C-PERSON}-OBJ)))
(triggered (heart-IN ({C-LOCATION}-OF)))	(triggered ({C-DATE}-ADV)(injuring-ADV))
(triggered (killing-ADV ({C-PERSON}-OBJ)))	(triggered (explosion-OBJ)(killing ({C-PERSON}-OBJ)))
(triggered ({C-DATE}-ADV))	...

Figure 1: (a) Example sentence on terrorism scenario. (b) Dependency Tree of the example sentence (The entities to be extracted are shaded in the tree). (c) Predicate-Argument patterns and Chain-model patterns that contribute to the extraction task. (d) Subtree model patterns that contribute the extraction task.

creases. In particular, many of the pattern candidates overlap one another. For a given set of extraction patterns, if pattern A subsumes pattern B (say, A is (shoot ({C-PERSON}-OBJ)(to death)) and B is (shoot ({C-PERSON}-OBJ))), there is no added contribution for extraction by pattern matching with A (since all the matches with pattern A must be covered with pattern B). Therefore, we need to pay special attention to the ranking function for pattern candidates, so that patterns with more relevant contexts get higher score.

3 Acquisition Method

This section discusses an automatic procedure to learn extraction patterns. Given a narrative description of the scenario and a set of source documents, the following three stages obtain the relevant extraction patterns for the scenario; *preprocessing*, *document retrieval*, and *ranking pattern candidates*.

3.1 Stage 1: Preprocessing

Morphological analysis and Named Entities (NE) tagging are performed at this stage.[7] Then all the sentences are converted into dependency trees by an appropriate dependency analyzer.[8] The NE tagging

[7] We used Extended NE hierarchy based on (Sekine et al., 2002), which is structured and contains 150 classes.

[8] Any degree of detail can be chosen through entire procedure, from lexicalized dependency to chunk-level dependency. For the following experiment in Japanese, we define a node in

replaces named entities by their class, so the resulting dependency trees contain some NE class names as leaf nodes. This is crucial to identifying common patterns, and to applying these patterns to new text.

3.2 Stage 2: Document Retrieval

The procedure retrieves a set of documents that describe the events of the scenario of interest, the *relevant document set*. A set of narrative sentences describing the scenario is selected to create a query for the retrieval. Any IR system of sufficient accuracy can be used at this stage. For this experiment, we retrieved the documents using CRL's stochastic-model-based IR system (Murata et al., 1999).

3.3 Stage 3: Ranking Pattern Candidates

Given the dependency trees of parsed sentences in the *relevant document set*, all the possible subtrees can be candidates for extraction patterns. The ranking of pattern candidates is inspired by TF/IDF scoring in IR literature; a pattern is more relevant when it appears more in the *relevant document set* and less across the entire collection of source documents.

The right-most expansion base subtree discovery algorithm (Abe et al., 2002) was implemented to calculate *term frequency* (raw frequency of a pattern) and *document frequency* (the number of documents where a pattern appears) for each pattern candidate. The algorithm finds the subtrees appearing more frequently than a given threshold by constructing the subtrees level by level, while keeping track of their occurrence in the corpus. Thus, it efficiently avoids the construction of duplicate patterns and runs almost linearly in the total size of the maximal tree patterns contained in the corpus.

The following ranking function was used to rank each pattern candidate. The score of subtree i, $score_i$, is

$$score_i = tf_i \cdot \left(\log \frac{N}{df_i}\right)^{\beta} \quad (1)$$

where tf_i is the number of times that subtree i appears across the documents in the *relevant document set*, R. T_R is the set of subtrees that appear in R. df_i is the number of documents in the collection containing subtree i, and N is the total number of documents in the collection. The first term roughly corresponds to the *term frequency* and the second term to the *inverse document frequency* in TF/IDF scoring. β is used to control the weight on the IDF portion of this scoring function.

the dependency tree as a *bunsetsu*, phrasal unit.

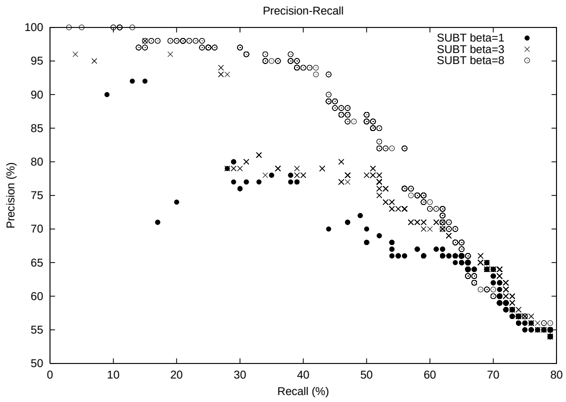


Figure 2: Comparison of Extraction Performance with Different β

3.4 Parameter Tuning for Ranking Function

The β in Equation (1) is used to parameterize the weight on the IDF portion of the ranking function. As we pointed out in Section 2, we need to pay special attention to overlapping patterns; the more relevant context a pattern contains, the higher it should be ranked. The weight β serves to focus on how specific a pattern is to a given scenario. Therefore, for high β value, (triggered (explosion-OBJ)({C-DATE}-ADV)) is ranked higher than (triggered ({C-DATE}-ADV)) in the terrorism scenario, for example. Figure 2 shows the improvement of the extraction performance by tuning β on the entity extraction task which will be discussed in the next section.

For unsupervised tuning of β, we used a pseudo-extraction task, instead of using held-out data for supervised learning. We used an unsupervised version of the text classification task to optimize β, assuming that all the documents retrieved by the IR system are relevant to the scenario and the pattern set that performs well on the text classification task also works well on the entity extraction task.

The unsupervised text classification task is to measure how close a pattern matching system, given a set of extraction patterns, simulates the document retrieval of the same IR system as in the previous sub-section. The β value is optimized so that the cumulative performance of the precision-recall curve over the entire range of recall for the text classification task is maximized.

The document set for text classification is composed of the documents retrieved by the same IR system as in Section 3.2 plus the same number of documents picked up randomly, where all the documents are taken from a different document set from the one used for pattern learning. The pattern matching system, given a set of extraction patterns, classifies a document as *retrieved* if any of the patterns match any portion of the document, and as *random* otherwise. Thus, we can get the performance of text classification of the pattern matching system in the form of a precision-recall curve, without any supervision.

Next, the *area* of the precision-recall curve is computed by connecting every point in the precision-recall curve from 0 to the maximum recall the pattern matching system reached, and we compare the *area* for each possible β value. Finally, the β value which gets the greatest *area* under the precision-recall curve is used for extraction.

The comparison to the same procedure based on the precision-recall curve of the actual extraction performance shows that this tuning has high correlation with the extraction performance (Spearman correlation coefficient $r_s = 0.83$ with 2% confidence).

3.5 Filtering

For efficiency and to eliminate low-frequency noise, we filtered out the pattern candidates that appear in less than 3 documents throughout the entire collection. Also, since the patterns with too much context are unlikely to match with new text, we added another filtering criterion based on the number of nodes in a pattern candidate; the maximum number of nodes is 8.

Since all the slot-fillers in the extraction task of our experiment are assumed to be instances of the 150 classes in the extended Named Entity hierarchy (Sekine et al., 2002), further filtering was done by requiring a pattern candidate to contain at least one Named Entity class.

4 Experiment

The experiment of this study is focused on comparing the performance of the earlier extraction pattern models to the proposed Subtree Model (SUBT). The compared models are the direct predicate-argument model (PA)[9], and the Chain model (CH) in (Sudo et al., 2001).

The task for this experiment is entity extraction, which is to identify all the entities participating in relevant events in a set of given Japanese texts. Note that all NEs in the test documents were identified manually, so that the task can measure only how well extraction patterns can distinguish the participating entities from the entities that are not related to any events. This task does not involve grouping entities associated with the same event into a single template to avoid possible effect of merging failure on extraction performance for entities. We accumulated the test set of documents of two scenarios; the Management Succession scenario of (MUC-6, 1995), with a simpler template structure, where corporate managers assumed and/or left their posts, and the Murderer Arrest scenario, where a law enforcement organization arrested a murder suspect.

The source document set from which the extraction patterns are learned consists of 117,109 Mainichi Newspaper articles from 1995. All the sentences are morphologically analyzed by JUMAN (Kurohashi, 1997) and converted into dependency trees by KNP (Kurohashi and Nagao, 1994). Regardless of the model of extraction patterns, the pattern acquisition follows the procedure described in Section 3. We retrieved 300 documents as a *relevant document set*.

The association of NE classes and slots in the template is made automatically; *Person, Organization, Post* (slots) correspond to C-PERSON, C-ORG, C-POST (NE-classes), respectively, in the Succession scenario, and *Suspect, Arresting Agency, Charge* (slots) correspond to C-PERSON, C-ORG, C-OFFENCE (NE-classes), respectively, in the Ar-

[9]This is a restricted version of (Yangarber et al., 2000) constrained to have a single place-holder for each pattern, while (Yangarber et al., 2000) allowed more than one place-holder. However, the difference does not matter for the entity extraction task which does not require merging entities in a single template.

	Succession	Arrest
IR description (translation of Japanese)	Management Succession: Management Succession at the level of executives of a company. The topic of interest should not be limited to the promotion inside the company mentioned, but also includes hiring executives from outside the company or their resignation.	A relevant document must describe the arrest of the suspect of murder. The document should be regarded as interesting if it discusses the suspect under suspicion for multiple crimes including murder, such as murder-robbery.
Slots	*Person*, *Organization*, *Post*	*Arresting Agency*, *Suspect*, *Charge*
# of Test Documents (relevant + irrelevant)	148 $(87 + 61)$	205 $(105 + 100)$
Slots	Person: 135 Organization: 172 Post: 215	Arresting Agency: 128 Suspect: 129 Charge: 148

Table 1: Task Description and Statistics of Test Data

rest scenario. [10]

For each model, we get a list of the pattern candidates ordered by the ranking function discussed in Section 3.3 after filtering. The result of the performance is shown (Figure 3) as a precision-recall graph for each subset of top-n ranked patterns where n ranges from 1 to the number of the pattern candidates.

The test set was accumulated from Mainichi Newspaper in 1996 by a simple keyword search, with some additional *irrelevant documents*. (See Table 1 for detail.)

Figure 3(a) shows the precision-recall curve of top-n relevant extraction patterns for each model on the Succession Scenario. At lower recall levels (up to 35%), all the models performed similarly. However, the precision of Chain patterns dropped suddenly by 20% at recall level 38%, while the SUBT patterns keep the precision significantly higher than Chain patterns until it reaches 58% recall. Even after SUBT hit the drop at 56%, SUBT is consistently a few percent higher in precision than Chain patterns for most recall levels. Figure 3(a) also shows that although PA keeps high precision at low recall level it has a significantly lower ceiling of recall (52%) compared to other models.

Figure 3(b) shows the extraction performance on the Arrest scenario task. Again, the Predicate-Argument model has a much lower recall ceiling (25%). The difference in the performance between the Subtree model and the Chain model does not seem as obvious as in the Succession task. However, it is still observable that the Subtree model gains a few percent precision over the Chain model at recall levels around 40%. A possible explanation of the subtleness in performance difference in this scenario is the smaller number of contributing patterns compared to the Succession scenario.

5 Discussion

One of the advantages of the proposed model is the ability to capture more varied context. The Predicate-Argument model relies for its context on the predicate and its direct arguments. However, some Predicate-Argument patterns may be too general, so that they could be applied to texts about a different scenario and mistakenly detect entities from them. For example, (({C-ORG}-SBJ) *happyo-suru*), "{C-ORG} reports" may be the pattern used to extract an *Organization* in the Succession scenario but it is too general — it could match irrelevant sentences by mistake. The proposed Subtree Model can acquire a more scenario-specific pattern (({C-ORG}-SBJ)((*shunin-suru*-REL) *jinji*-OBJ) *happyo-suru*) "{C-ORG} reports a personnel affair to appoint". Any scoring function that penalizes the generality of a pattern match, such as inverse document frequency, can successfully lessen the significance of too general patterns.

[10] Since there is no subcategory of C-PERSON to distinguish *Suspect* and victim (which is not extracted in this experiment) for the Arrest scenario, the learned pattern candidates may extract victims as *Suspect* entities by mistake.

(a) (b)

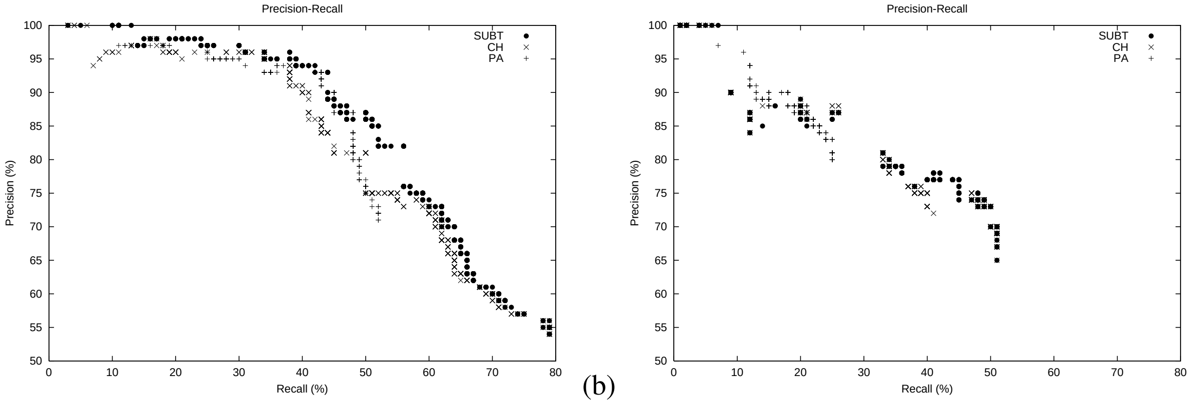


Figure 3: Extraction Performance: (a) Succession Scenario ($\beta = 8$), (b) Arrest Scenario ($\beta = 4$)

The detailed analysis of the experiment revealed that the overly-general patterns are more severely penalized in the Subtree model compared to the Chain model. Although both models penalize general patterns in the same way, the Subtree model also promotes more scenario-specific patterns than the Chain model. In Figure 3, the large drop was caused by the pattern (({C-DATE}-ON) {C-POST}), which was mainly used to describe the date of appointment to the C-POST in the list of one's professional history (which is not regarded as a Succession event), but also used in other scenarios in the business domain (18% precision by itself). Although the scoring function described in Section 3.3 is the same for both models, the Subtree model can also produce contributing patterns, such as (({C-PERSON}{C-POST}-SBJ)({C-POST}-TO) *shunin-suru*) "{C-PERSON}{C-POST} was appointed to {C-POST}" whose ranks were higher than the problematic pattern.

Without generalizing case marking for nominalized predicates, the Predicate-Argument model excludes some highly contributing patterns with nominalized predicates, as some example patterns show in Figure 4. Also, chains of modifiers could be extracted only by the Subtree and Chain models. A typical and highly relevant expression for the Succession scenario is (((*daihyo-ken*-SBJ) *aru*-REL) {C-POST}) "{C-POST} with ministerial authority".

Although, in the Arrest scenario, the superiority of the Subtree model to the other models is not clear, the general discussion about the capability of capturing additional context still holds. In Figure 4, the short pattern (({C-PERSON}{C-POST}-APPOS) {C-NUM}), which is used for a general description of a person with his/her occupation and age, has relatively low precision (71%). However, with more relevant context, such as "arrest" or "unemployed", the patterns become more relevant to Arrest scenario.

6 Conclusion and Future Work

In this paper, we explored alternative models for the automatic acquisition of extraction patterns. We proposed a model based on arbitrary subtrees of dependency trees. The result of the experiment confirmed that the Subtree model allows a gain in recall while preserving high precision. We also discussed the effect of the weight tuning in TF/IDF scoring and showed an unsupervised way of adjusting it.

There are several ways in which our pattern model may be further improved. In particular, we would like to relax the restraint that all the fills must be tagged with their proper NE tags by introducing a GENERIC place-holder into the extraction patterns. By allowing a GENERIC place-holder to match with anything as long as the context of the pattern is matched, the extraction patterns can extract the entities that are not tagged properly. Also patterns with a GENERIC place-holder can be applied to slots that are not names. Thus, the acquisition method described in Section 3 can be used to find the patterns for any type of slot fill.

[11]({C-POST} is used as a title of {C-PERSON} as in *President* Bush.)

Pattern	Correct	Incorrect	SUBT	Chain	PA
(({C-PERSON}{C-POST}-OF) {C-PERSON} *shoukaku*)	26	1	yes	yes	no
promotion of {C-POST}{C-PERSON}[11]					
(((*daihyo-ken*-SBJ) *aru*-REL) {C-POST})	4	0	yes	yes	no
{C-POST} with ministerial authority					
((((*daihyo-ken*-(*no*)SBJ) *aru*-REL) {C-POST}-TO) *shunin-suru*)	2	0	yes	yes	no
be appointed to {C-POST} with ministerial authority					
(({C-ORG}-SBJ) *happyo-suru*)	16	3	yes	yes	yes
{C-ORG} reports					
(({C-ORG}-SBJ) (*jinji*-OBJ) *happyo-suru*)	4	0	yes	no	yes
{C-ORG} report personnel affair					
(({C-PERSON}{C-POST}-APPOS) {C-NUM})	54	22	yes	yes	no
((({C-PERSON}{C-POST}-APPOS) {C-NUM}) *taiho-suru*	17	1	yes	yes	no
arrest {C-PERSON}{C-POST}, {C-NUM}					
(((*mushoku*-APPOS){C-PERSON}{C-POST}-APPOS) {C-NUM})	11	0	yes	yes	no
{C-PERSON}{C-POST}, {C-NUM}, unemployed					

Figure 4: Examples of extraction patterns and their contribution

Acknowledgments Thanks to Taku Kudo for his implementation of the subtree discovery algorithm and the anonymous reviewers for useful comments. 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.

References

Kenji Abe, Shinji Kawasoe, Tatsuya Asai, Hiroki Arimura, and Setsuo Arikawa. 2002. Optimized Substructure Discovery for Semi-structured Data. In *Proceedings of the 6th European Conference on Principles and Practice of Knowledge in Databases (PKDD-2002)*.

Sadao Kurohashi and Makoto Nagao. 1994. KN Parser : Japanese Dependency/Case Structure Analyzer. In *Proceedings of the Workshop on Sharable Natural Language Resources*.

Sadao Kurohashi, 1997. *Japanese Morphological Analyzing System: JUMAN*. http://www.kc.t.u-tokyo.ac.jp/nl-resource/juman-e.html.

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

Masaki Murata, Kiyotaka Uchimoto, Hiromi Ozaku, and Qing Ma. 1999. Information Retrieval Based on Stochastic Models in IREX. In *Proceedings of the IREX Workshop*.

Ellen Riloff and Rosie Jones. 1999. Learning Dictionaries for Information Extraction by Multi-level Bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*.

Ellen Riloff. 1993. Automatically Constructing a Dictionary for Information Extraction Tasks. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*.

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

Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. 2002. Extended Named Entity Hierarchy. In *Proceedings of Third International Conference on Language Resources and Evaluation (LREC 2002)*.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2001. Automatic Pattern Acquisition for Japanese Information Extraction. In *Proceedings of the Human Language Technology Conference (HLT2001)*.

Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. 2000. Unsupervised Discovery of Scenario-Level Patterns for Information Extraction. In *Proceedings of 18th International Conference on Computational Linguistics (COLING-2000)*.