Unsupervised Discovery of Extraction Patterns for Information Extraction

by

Kiyoshi Sudo

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

New York University

September 2004

To my wife and my mother.

ACKNOWLEDGEMENTS

First of all, I would like to thank Professor Ralph Grishman and Professor Satoshi Sekine for their patient and generous support throughout my academic career. I was lucky to have two such prominent professors as my research advisers. I learned a lot from Ralph's cautiousness and Satoshi's ambition. Without either of them, any of this work could not have been done.

This research was supported in part by the Defense Advanced Research Projects Agency as part of the Translingual Information Detection, Extraction and Summarization (TIDES) program, under Grant N66001-001-1-8917 from the Space and Naval Warfare Systems Center, San Diego, and by the National Science Foundation under Grant IIS-0081962 and IIS-00325657. This paper does not necessarily reflect the position of the U.S. Government.

I thank Professor I. Dan Melamed for his encouragement and openness as . I am also glad to have had Professor Yann LeCun and Professor Dennis Shasha in my thesis committee.

I thank my family members both in Japan and in New York and many friends around the world for always reminding me that I am never without their love and supports.

Lastly, I would not have been able to make it through all the years without care and support from Min Young.

ABSTRACT

The task of Information Extraction (IE) is to find specific types of information in natural language text. In particular, *event extraction* identifies instances of a particular type of event or fact (a particular "scenario"), including the entities involved, and fills a database which has been pre-defined for the scenario. As the number of documents available on-line has multiplied, entity extraction has grown in importance for various applications, including tracking terrorist activities from newswire sources and building a database of job postings from the Web, to name a few.

Linguistic contexts, such as predicate-argument relationships, have been widely used as *extraction patterns* to identify the items to be extracted from the text. The cost of creating extraction patterns for each scenario has been a bottleneck limiting the portability of information extraction systems to different scenarios, although there has been some research on semi-supervised pattern discovery procedures to reduce this cost. The challenge is to develop a fully automatic method for identifying extraction patterns for a scenario specified by the user.

This dissertation presents a novel approach for the unsupervised discovery of extraction patterns for event extraction from raw text. First, we present a framework that allows the user to have a self-customizing information extraction system for his/her query: the Query-Driven Information Extraction (QDIE) framework. The input to the QDIE framework is the user's query: either a set of keywords or a narrative description of the event extraction task.

Second, we assess the improvement in extraction pattern models. By considering the shortcomings of the prior work based on predicate-argument models and their extensions, we propose a novel extraction pattern model that is based on arbitrary subtrees of dependency trees.

Third, we address the issue of portability across languages. As a case study of the QDIE framework, we implemented a pre-CODIE system, a Cross-Lingual On-Demand Information Extraction system requiring minimal human intervention, which incorporates the QDIE framework as a component for pattern discovery. In addition, we assess the role of machine translation in cross-lingual information extraction by comparing translation-based implementations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Information Extraction (IE) is a subfield of Natural Language Processing (NLP) that aims to extract specific types of information from text. Unlike information retrieval, which returns a set of documents for a given query, IE systems identify facts/events described in a text, and convert them into a structured format for ease of further processing, such as text mining or summarization.

> Japan launched its first H2-A rocket on Wednesday, putting a once-promising rocket program "back on track" after years of problems.

Figure 1.1: Japanese Rocket Launch Event

Date	Location	Type	Name	Status
Wednesday (8/29/2001)	Japan	Rocket	H2-A	Succeeded

Table 1.1: Example Table of Rocket Launch

Using an IE system, the user can get information about a rocket launch event as shown in Table 1.1 from the text in Figure 1.1. Note that a text may (and often does) contain supplemental information, such as background information. The main challenge of IE, therefore, is to identify the relevant information among the whole text.

1.1 Information Extraction and Domain Dependent Knowledge Base

For years, Message Understanding Conferences (MUCs) have been a standard evaluation of IE systems. In MUC, the user's need was explicitly defined as a type of event in a tabular form (scenario template task). The target event structure was shown in the form of a *template*, as exemplified in Table 1.2.

The recent MUC's "one-month rule" on system development reflects a typical IE system development. It states that the participants were allowed to

```
+ launch_event

       + vehicle_info

       + payload_info

       + launch_date (time)

       + launch_site (location)

       + mission_type (military, civilian)

       + mission_function (test, deploy, retrieve)

       + mission_status (succeeded, failed, in-progress,

scheduled)
```

Table 1.2: Rocket Launch Event Template

have a one month period to develop their IE system given the definition of the task and training data (a set of text annotated with the extracted entities). Once given the task and its template, the developers of the IE system identified the sentence patterns which contain the information of the target events; such patterns are called *extraction patterns.* For example, in the terrorism domain used in MUC-4, a possible pattern would be formed as "<*target*> was bombed." This pattern could be applied onto the sentence "A public building was bombed in Oklahoma City last year." and "A public building" can be extracted as *target*. Thus, if we were informed beforehand about the event, event structure, and its domain, we could develop an IE system well customized for the particular target event structure.

However, because of the wide availability of on-line documents and the diversity of needs for information extraction, it seems impractical to develop an IE system for each task. A recent focus of the studies of information extraction is toward its portability across *domains*. Having the portability across domains, an IE system can relatively easily be customized to a new domain. Thus, faster development of IE systems is possible.

1.2 Pattern Discovery

The cost of pattern discovery varies as the degree to which human labor is involved. Here is the comparison on issues of pattern discovery, when patterns are created *manually*, *semi-automatically*, *automatically*.

- *Manual*

  In earlier MUCs, many IE systems were customized manually to a given task. [5, 11, 15] However, the costs are prohibitively high to devote computational linguists for building up and maintaining knowledge-bases for each domain of request.

- *Semi-Automatic*

  A semi-automatic discovery procedure of extraction patterns is not fully automated, but aims to help the user create the knowledge-bases, such as lexicons and patterns [39]. The methodology to assist the user includes showing a pattern candidate and incorporating an existing ontology and pattern set.

- *Automatic*

  An automatic procedure for pattern discovery takes a set of documents and outputs a set of extraction patterns, by using a machine learning technique. Among automatic pattern discovery methods, there is a wide range of preparation cost for training data, which will be covered in detail in Chapter 2;

  - *Supervised learning systems* cannot ignore the cost of a large amount of annotated data [25, 14, 34, 12, 33]

  - *Semi-supervised learning systems*, most of which are based on a bootstrapping method, require only a handful of initial data to acquire. [28, 40, 4]

– *Unsupervised learning systems*, further reduce the burden on the user to require only a statement of the user's information need. Since no extraction patterns are given in advance by the user, the main obstacle is to realize the user's need into a set of extraction patterns, which is the central theme of this dissertation.

1.3 Extraction Patterns

Before discussing the issues of pattern discovery in detail, we would like to formalize the definition of extraction patterns and the problem of pattern discovery.

This dissertation will focus on extraction patterns for information extraction. For many information extraction systems, extraction patterns are designed to match a portion of text and assign the slot to the matched entity. In the following, we will explain extraction pattern, dependency tree, and pattern discovery.

```
(PRED: trigger (SBJ: A Palestinian suicide bomber)
               (OBJ: a massive explosion)
               (IN:  in (MOD: the heart)
                        (IN:   of (MOD: downtown [LOC: Jerusalem]))
               (ADV: today))
```

Figure 1.2: Example of Extraction Pattern

Definition: (Extraction pattern) An *extraction pattern* is a dependency tree one of whose nodes is a *place holder*. The rest of the nodes in an extraction pattern are called the *context* of the pattern. An extraction pattern can *match* with a portion of sentence if all the nodes in the extraction pattern can match with the nodes in the sentence portion.

Context nodes are lexicalized nodes that can match only with the node with the same label. The *place holder* node contains a Named Entity class that matches any instance of the same class. Having an *assignment rule* from an extraction pattern to a *slot* in the template, the entity matched with a *place holder* is extracted as an output slot-filler.

Definition: (Dependency Tree) A *dependency relationship* is an asymmetric binary relationship between a word called the head (or governor, parent), and another word called the modifier (or dependent, daughter). Depen-

SBJ	Subject
OBJ	Object
IN	preposition
MOD	others

Table 1.3: Types of Dependency Relations

dency grammars represent sentence structures as a set of dependency relationships. Normally the dependency relationships form a tree that connects all the words in a sentence. A word in the sentence may have several modifiers, but each word may modify at most one word. The root of the dependency tree does not modify any word. It is also called the head of the sentence.

Table 1.3 shows the types of dependency relations used in this research. Figure 1.3 gives an example of an S-expression of the dependency structure of a sentence. In practice, all the dependencies given for a sentence will be acquired automatically by a dependency parser.

Definition: (Pattern Discovery) *Pattern discovery* is the task of identifying the extraction patterns that are relevant for the user's need, specified by the user's *query*. Given a source text where all the sentences are parsed into

```
(PRED: trigger (SBJ: A Palestinian suicide bomber)
               (OBJ: a massive explosion)
               (IN:  in (MOD: the heart)
                        (IN:   of (MOD: downtown [LOC: Jerusalem]))
               (ADV: today))
```

Figure 1.3: Example of Dependency Tree (in S-expression)

dependency trees, the task of pattern discovery is to identify the extraction patterns relevant to the query.

1.4 Contributions

The main contribution of this study is to provide a novel framework for unsupervised discovery of extraction patterns for event extraction tasks from raw text; the Query-Driven Information Extraction (QDIE) framework. This framework allows the user to have a self-customizing information extraction system for his/her query – it provides the portability to different domains.

We have assessed the improvement of extraction pattern models. By discussing the shortcomings of the prior work based on predicate-argument models and its extension, we proposed a novel extraction pattern model that is based on arbitrary subtrees of dependency trees. We showed that the new subtree model performed best among different pattern models. We also discussed some techniques needed for pattern discovery of the subtree model, where the number of candidate patterns is larger than the prior work.

As a case study of QDIE framework, we have implemented pre-CODIE (Cross-Lingual On-Demand IE) system, a cross-lingual on-demand information extraction system with minimal human intervention, which incorporates QDIE framework as a component of pattern discovery. In addition, we have assessed the role of machine translation in cross-lingual information extraction by comparing translation-based implementations.

1.5 Organization of Thesis

The following chapters of the thesis are organized as follows. Chapter 2 surveys the previous pattern discovery work, mainly focused on semi- and unsupervised learning methods. Chapter 3 overviews our novel pattern discovery procedure: Query-Driven Information Extraction. Chapter 4 discusses the improved models for extraction patterns. Chapter 5 illustrates a case-study of QDIE framework with the discussion of the role of machine translation in cross-lingual information extraction system. Finally, we conclude the dissertation with prospects of the current work in Chapter 6.

Chapter 2

Prior Work

This chapter introduces the prior work on automatic learning of extraction patterns for information extraction. We discuss learning methods, the model for extraction patterns, and the amount of engineering effort for each method. First, the supervised learning methods that need annotated data are presented. Then, the semi-supervised learning methods are reviewed.

2.1 Supervised

In the approaches with supervised learning for pattern discovery, the template for an extraction task is given beforehand. Most of the methods take the parts of the sentences which are annotated as containing a slot-filler as the most specialized instances of extraction patterns. Then the patterns are generalized to have a wide coverage in some way.

2.1.1 Riloff 93

The AutoSlog system [25] extracts a dictionary of what they called *conceptual nodes* from annotated documents. A conceptual node is an extraction rule which includes a trigger word to be activated. When the trigger word is matched in the text and the conditions that the concept node specifies are satisfied, the concept node is activated and the slot in the concept node definition is extracted from the context.

In the training phase, AutoSlog identifies a sentence annotated with a slot filler and semantic tag. Then, AutoSlog looks up its heuristic rule list, a part of which is shown in Table 2.1, and sees if any of the rules can match the part of the sentence that contains the slot filler.

For example, in the terrorist domain used in MUC-4, given a sentence, "..., public buildings were bombed and a car-bomb was detonated" that is annotated with the slot filler for <target> slot, "public buildings", AutoSlog would generate from the heuristic, <subject> passive-verb, <target> was bombed,

<subject> passive-verb
<subject> active-verb
passive-verb <dobj>
active-verb <dobj>
noun prep <np>

Table 2.1: Sample of AutoSlog Heuristic Rules

with the word "bombed" as the trigger word of this rule.

Each rule handles only a single-slot extraction and the later stage of the whole IE system [19] re-assembled the slots to build up an event structure possibly with more than one slot. AutoSlog also used a semantic tagger incorporated with their whole IE system, and used semantic constraints in the extraction rule. Unlike later work, AutoSlog does not relax constraints or merge similar concept nodes to generalize. Thus, it depends on good training data with annotation of the sentence which carries the target information and the slot filler phrase.

2.1.2 Kim and Moldovan 95

The PALKA [14] system produces the extraction rule as a pair of a meaning frame and a phrasal pattern, called *Frame-Phrasal pattern structure (FP-structure)*. Each training phase tries to apply the extraction rules that have been learned in the previous iterations, and if it fails it creates a new rule and tries to generalize it with existing ones, or if it generates the wrong interpretation it specializes the existing rules.

PALKA uses a concept hierarchy to generalize or specialize the extraction rules. The way PALKA generalizes the semantic constraints is based on inductive learning mechanism [23, 22] to move up the semantic hierarchy so that the extraction rules can cover as many positive examples as possible, and it specializes the constraints so that the rules will not include any negative examples.

Thus, PALKA is claimed to be sensitive to noise of training data and a single negative instance leads to stopping a possible good generalization. It also needs a existing semantic hierarchy to generalize and specialize the constraints.

2.1.3 Soderland et al. 95

The CRYSTAL system [34] takes texts that are processed by a syntactic parser. For its training phase, CRYSTAL needs training documents annotated by a domain expert, as well as the semantic hierarchy for medical domain. CRYSTAL starts learning by creating the extraction rules for each instance of the target event in the training data. Then it finds the most similar pair of extraction rules and merges them together by finding the most restrictive constraints that cover both rules. This is a form of inductive learning [23].

CRYSTAL does not extract the exact phrase for a slot filler but only points out the syntactic field that contains the slot filler, as AutoSlog [25] does.

2.1.4 Huffman 96

The LIEP system [12] allowed user interaction for identifying events in texts, based on the assumption that a large annotated training corpus is hard to obtain.

Given a training set of texts with the information of event type and slot fillers, first, LIEP finds a syntactic relationship between the slot fillers in the event instance. Once it finds out the plausible relationship, LIEP creates a new pattern for this particular instance. For generalization, if LIEP finds two patterns with the same syntactic structure but different lexical constituents, it creates a pattern with the same syntactic structure but a generalized class for its constituents, as shown in Figure 2.1. The generalized pattern is tested on the training corpus and, if it performs better than two separate patterns, LIEP puts the generalized pattern into its pattern dictionary.

```
        PERSON be named TITLE of COMPANY

      PERSON be appointed TITLE at COMPANY

                      ⇓

PERSON be (named, appointed) TITLE (of, at) COMPANY
```

Figure 2.1: Sample Patterns and their Generalization

2.1.5 Soderland 99

The WHISK [33] system learns extraction patterns. The training data is a set of sentences marked as containing an event instance and a list of the slot fillers of the template that are contained in the sentence. WHISK creates a rule from the event instance which will identify the boundaries of the slot fillers in the sentence.

```
Pattern:

   * (<person>) * Succeeds * (<person>) * (<company>)

Output Rule:

   Succession {PsnIN $1} {PsnOUT $2} {Org $3}

Sentence:

   '<Mr.A> succeeds <Mr.B>, chairman of <XYZ Inc.>'

Output:

   Succession {PsnIN Mr.A} {PsnOUT Mr.B} {Org XYZ Inc.}
```

Figure 2.2: WHISK Rule Creation

WHISK extends a rule by adding terms testing the performance of the extended pattern on the training set.

2.1.6 Discussion

The main bottleneck of the supervised pattern discovery methods is the cost of preparation of the training data. Many of the surveyed systems in this section need a large amount of annotated document for a particular extraction task. The cost for manual annotation on the documents can not be ignored. This also leads to the lack of portability of an IE system; one needs to get the annotated data for each task.

Moreover, many of the surveyed systems assumed the use of a semantic tagger to mark the expressions of some semantic classes, such as person, location and organization names. Although the Named Entities detection task in the Message Understanding Conferences [1, 2] has been well explored and the performance that the NE systems achieved is high, there are many other types of semantic classes especially when the target domain is specific, such as disease names for medical domain and weapon names for terrorism domain. The semantic classes necessary for a particular task are restricted and, therefore, a general knowledge base, such as an existing thesaurus or dictionary, does not have wide enough coverage.

2.2 Towards Unsupervised

It has been pointed out that it took a lot of labor to prepare the training data for an information extraction system. The idea of cost reduction leads to the approaches of unsupervised learning. The work described in this section is based on bootstrapping methods, expanding an initial small set of extraction patterns. In that sense, we call these works semi-supervised methods.

2.2.1 Riloff 96

Cost reduction for knowledge extraction has advanced further since Riloff's original automatic pattern discovery system AutoSlog [25] was built. AutoSlog-TS [26] needs only a corpus pre-classified with regard to each document's relevance to the topic of the task. AutoSlog-TS introduced a measurement of document relevance to the task.

The input to AutoSlog-TS is a document set pre-classified with respect to relevance to the task of interest. First, AutoSlog-TS applies heuristic rules exhaustively to the training corpus and produces a list of possible extraction patterns.

Then we can calculate the score of each pattern p in the set of the possible extraction patterns P.

Define $f_R(p)$ as the frequency of the pattern p being activated in the relevant documents and $f(p)$ as the frequency of p being activated in any document.

$$score(p) = \frac{f_R(p)}{f(p)} \cdot \log(f(p))$$

Here, $\frac{f_R(p)}{f(p)}$ is $Pr(\text{relevant text}|\text{text contains } p)$, which measures the conditional probability that a text is relevant given the text contains a certain extraction pattern. It aims to find domain-specific patterns that are likely to appear in the relevant document set. Also, there is a compromise between the frequency of the pattern in general and the frequency only in the relevant document set by using less weighted (logarithm of) frequency.

2.2.2 Riloff and Jones 99

Mutual bootstrapping is one form of co-training used in Riloff and Jones [28] for lexical discovery. Lexicons and extraction patterns are used as two separate features. Given a handful of lexical entries as initial data, the system finds patterns that extract the initial lexicon. The extracted patterns are ranked and the most reliable patterns are used to extract more lexical items.

The assumption of using mutual bootstrapping is the duality that a good pattern can find a good lexicon and a good lexicon can find a good pattern. Thus, the a set of lexicons would find the extraction patterns that can reliably extract them and, in turn, the extraction patterns can find another set of lexicons, while keeping the relevancy of the lexicons.

The algorithm of mutual bootstrapping is shown in Figure 2.3.

```
Given:
   A set of lexicons $L$ with seed words
   A set of patterns $P$ currently empty
   A set of candidate extraction patterns
exhaustively extracted by $AutoSlog$
Loop:
   Score all candidate extracted patterns.
   Take the highest extraction pattern as $best$.
   Add $best$ to $P$.
   Add lexicons of $best$'s extraction to $L$.
```

Figure 2.3: Mutual Bootstrapping

The system scores all the patterns extracted by AutoSlog [25] (See Section 2.1.1) that exhaustively takes all possible extraction pattern. The algorithm takes the best-ranked pattern p into the set P of the extracted patterns and then add the lexicon that is extracted by p into L. The mutual bootstrapping repeats starting with scoring all the patterns with the updated L.

A caveat of mutual bootstrapping approach is that a minor error would cause a large amount of error during the next iteration. For example, Riloff and Jones reported that, for the extraction task of location names in the terrorist domain, mutual bootstrapping wrongly extracted the non-location phrases, such as *head, clash, air*, from the reasonable extraction pattern "shot in $< x >$"

Another level of bootstrapping, called *meta-bootstrapping*, was introduced as a remedy of the problem above, shown in Figure 2.4. In meta-bootstrapping, each iteration takes only the 5 best NPs for adding to the extracted lexicons, thus trying to be conservative rather than taking all the lexical entries that are extracted by the patterns.

2.2.3 Collins and Singer 99

One of the interesting applications of co-training to an IE task was done by Collins and Singer [9]. They take the NE task as a classification problem of noun phrases, where the goal is to learn the hypothesis to classify noun phrases given a set of their features.

The NE task was defined to find a classifier f given a feature vector x. All the instances of named entities are described with two set of features; a *spelling* feature set, such as the full string of the phrase and the feature whether

```
Given:
   A set of initial lexicons
   A set of lexicons $L$ as ``permanent semantic
lexicon'' (final answer)
Loop:
   Initialize a temporary lexicon set $TmpL$
for mutual bootstrapping
   Run mutual bootstrapping with $TmpL$.
   Add 5 best NPs from the result $TmpL$ from
mutual bootstrapping.
```

Figure 2.4: Meta-bootstrapping

it contains a particular string, and *context* feature set, such as the context itself and the type of the context. Thus, all the instances have two *views* in terms of [7].

One form of the application to the NE task Collins and Singer described was a decision-list based co-training algorithm. The algorithm, called DL-Cotrain, first creates a decision list of the *spelling* features from the seed words and labeled the training data. Then, DL-CoTrain algorithm set a decision list of the *context* features from the newly labeled instances in the training data, so far. The training data is labeled again with the decision list of *context* features. Thus, the part of the training data which was not labeled before is now added to the labeled data, so the decision lists are learned from the larger set than the previous iteration.

Another form of co-training based algorithm is CoBoost algorithm. CoBoost algorithm is based on boosting algorithm. It tries to minimize the disagreement on the training data of two classifiers which are based on two different *views*.

2.2.4 Yangarber 00

EXDISCO [40] is another interesting extraction strategy with mutual bootstrapping. The motivation behind the mutual bootstrapping of EXDISCO is the circularity where the presence of the relevant documents indicates good patterns and good patterns can find relevant documents .

EXDISCO requires only an unannotated corpus and a handful of seed patterns. First the document set is divided into a *relevant document set* that contains at least one instance of patterns and *non-relevant document set* that do not contain any seed patterns. Then the candidate patterns are generated from the clauses in the documents and ranked in correlation with the relevant documents.

The score of pattern *pt* is calculated by

$$score(pt) = \frac{doc_R(pt)}{doc(pt)} \cdot \log(doc_R(pt))$$

where $doc_R(pt)$ is the number of relevant documents that contain *pt*.

EXDISCO adds the highest pattern to the pattern set that initially contained only seed patterns. Then, EXDISCO re-ranks each document using the newly obtained pattern set.

At the (i+1)-th iteration, the score of each document d is calculated by,

$$score^{i+1}(d) = \max\left(score^i(d), \frac{1}{|DOC(PT_d)|} \cdot \sum_{d \in DOC(PT_d)} score^i(d)\right)$$

where PT_d is a set of patterns that match document d and $DOC(PT_d)$ is a set of documents that all patterns in PT_d match. This score calculates the sum of the normalized scores of the documents which contained all the patterns that now are found in the document in question. Thus, having all the documents updated score, EXDISCO again splits the entire set of documents into *relevant* and *non-relevant* and keeps iterating.

2.2.5 Agichtein and Gravano 00

Local relations between entities are learned in the work of Agichtein and Gravano [4]. Agichtein and Gravano's Snowball system for extracting relations between location and organization is based on Dual Iterative Pattern Expansion (DIPRE) algorithm [8]. DIPRE, similar to co-training, works well on data that have two distinct features, each of which can independently distinguish the target class of instances from the others. The main algorithm of Snowball, first, induces and ranks the extraction patterns given a handful of initial relation instances; a set of pairs of location and organization. The algorithm applies the patterns onto the source text and tries to find other relations that the patterns can cover. From the newly found relations, the patterns are re-ranked.

The metrics of the patterns and relations used for each iteration is important since a tiny error would extract more erroneous patterns/relations in the later stages. The confidence of a pattern is calculated by similar method to Riloff [26], where the score is a multiplication of the relevance of the pattern (precision) and the logarithm of the frequency of the pattern. Then, the confidence of each relation is calculated by the probability that all the patterns that extract the relation were triggered incorrectly.

2.2.6 Discussion

The drawback of this type of bootstrapping method is that it is prone to take tiny errors and amplify them during iteration. In fact, this problem is caused not only by noise but also (more seriously) by polysemous words and phrases. For example, although the pattern "kidnapped in $< x >$" in terrorist domain itself is a good pattern, it will extract not only locations "kidnapped in *Tokyo*" but also many dates "kidnapped in *January.*"

We can impose an NE category constraint for a matching variable to reduce the degree of polysemous patterns. Thus, the example patterns above are expressed as two different patterns, "kidnapped in {C-LOC}" and "kidnapped in {C-DATE}".

In supervised pattern discovery methods, the cost of preparing the training data was high, annotating all the necessary information. The work surveyed in this section tries to reduce the burden of annotating the training data. Riloff reported in [27] that it took a user 8 hours to annotate 160 texts, roughly a week to annotate 1000 texts for her previous work AutoSlog [25]. AutoSlog-TS [26] takes a corpus pre-classified for relevancy as initial input. Classification of documents is a far easier task than making annotations although the actual time it took was not reported. Furthermore, by using mutual bootstrapping approach, the cost reduction on the training data were further pursued. Snowball and EXDISCO required only an unannotated corpus and a handful of the initial relations and patterns, respectively.

However, the number of documents relevant to the task is limited because of the degree of specificity of the extraction task. The data-sparseness problem must be considered especially when the task is very specific and the source data is general.

From the user's point of view, providing a small set of extraction patterns may still be a bottleneck. In Chapter 3, we will discuss more about the bottleneck and need for a fully unsupervised method for pattern discovery.

Chapter 3

Overview of Pattern Acquisition

3.1 Introduction

Chapter 2 surveyed various approaches to reduce the cost of porting an IE system into another domain. Especially, we discussed the discovery of extraction patterns, since today's IE systems are commonly based on pattern matching. Many of the previous approaches attempted to solve the pattern discovery problem as a classification problem where we separate relevant patterns from irrelevant ones – it reduced the cost of creating extraction patterns to the preparation of training data. Some work requires the user to provide only the relevance judgment of documents for pattern discovery. Significant reductions in human preparation cost may be achieved by employing bootstrapping algorithms for pattern discovery. With such a methodology, the user only needs to provide a small set of initial extraction patterns.

However, creating extraction patterns can be difficult in some situations. There are no standard criteria for choosing a "good" seed set of extraction patterns. Especially, it will be difficult in the cross-lingual IE case where the user has no knowledge of the source language. Chapter 5 will discuss more details of cross-lingual IE.

In this chapter we will propose a novel methodology of automatic pattern discovery where the user only needs to provide a query in the form of a narrative or a set of keywords – the same as used in information retrieval (IR) systems. To discover the extraction patterns based on the user's query, we realized an unsupervised pattern discovery framework incorporating an information retrieval system: Query-Driven Information Extraction (QDIE) framework. We employed an information retrieval system to get the documents relevant to the user's query. Subsequent processes transfer the relevance judgments into a set of extraction patterns, which will be discussed in the next section.

3.2 QDIE: Query-Driven Information Extraction

Unsupervised model: QDIE = IR + IE For minimizing the amount of human intervention, a set of keywords from the user used by information retrieval is an ideal candidate. We incorporated information retrieval into pattern discovery based on the following intuition; if a set of documents discusses the type of events that is relevant to the user's need, the extraction patterns used to describe the events in the documents are likely to appear more frequently in this set of documents, rather than to be spread across domains. Thus, we used TF/IDF-based scoring function for realizing that intuition and incorporated information retrieval and pattern discovery in a cascading fashion, as we will discuss in the next section.

In the following sections, we describe each component of the QDIE framework illustrated in Figure 3.1. First, the original documents are processed by a dependency parser and NE-tagger in advance. Then the system retrieves the relevant documents for the scenario as a *relevant document set*, specified by the user's query. Finally, from all the sentences in the *relevant document* *set*, the connected subtrees of dependency trees that conform to the definition of extraction patterns are considered as pattern candidates. The pattern candidates are ordered by the TF/IDF-based scoring function.

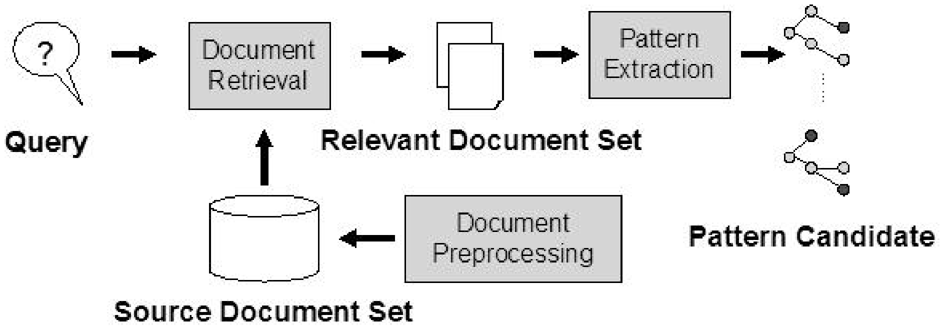


Figure 3.1: QDIE framework data-flow

3.2.1 Document Preprocessing

Dependency parsing and Named Entity (NE) tagging is performed on the training data at this stage. All NEs are represented as their class labels, and each sentence is represented as a dependency tree whose nodes correspond to a chunk. (Figure 3.2)

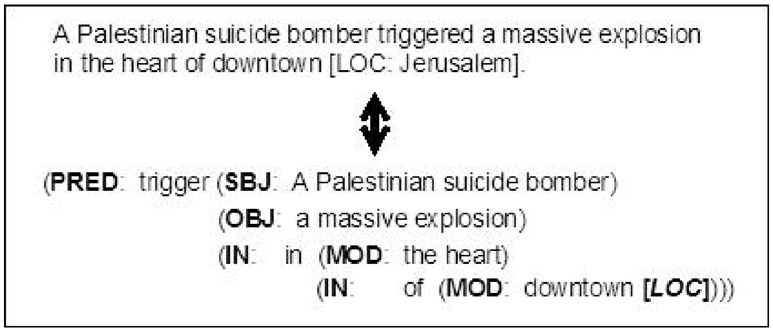

A Palestinian suicide bomber triggered a massive explosion in the heart of downtown [LOC: Jerusalem].
(PRED: trigger (SBJ: A Palestinian suicide bomber)
(OBJ: a massive explosion)
(IN: in (MOD: the heart)
(IN: of (MOD: downtown [LOC])))

Figure 3.2: Example of Sentence Preprocessing

3.2.2 Document Retrieval

As the first phase of our cascading procedure, a set of documents is retrieved in response to the user's query. Any modest information retrieval approach may suffice at this phase. All the sentences used to create the patterns are retrieved from this *relevant document set.*

The form of the query varies from a set of keywords (e.g. "airplane crash") to a narrative query (such as the TREC query: "A relevant document will include a prediction about the prime lending rate (national-level or major banks'), or will report a prime rate move by major banks, in response to or in anticipation of a federal/national-level action, such as a cut in the discount rate.").

3.2.3 Pattern Extraction

The dependency trees of the sentences from retrieved documents are the source of extraction pattern candidates. From each sentence, a set of dependency subtrees which conform to the pattern model (see Chapter 4) become *pattern candidates*. The QDIE system collects all the pattern candidates from retrieved sentences and calculates the relevance score for each candidate.

As illustrated in Figure 3.3, for each pattern i, the score of relevance, $score_i$ is calculated based on the TF/IDF calculation.

$$score_i = tf_i \cdot \log \frac{N}{df_i} \tag{3.1}$$

Here, tf_i denotes the number of times pattern candidate i occurred in the *relevant documents* in R, df_i denotes the number of documents in the entire source document set which contain pattern candidate i, and N is the number of documents in the source document set.

3.3 Conclusion

Query-Driven Information Extraction (QDIE) takes the user's query and source text as input and returns a set of extraction patterns relevant to the events

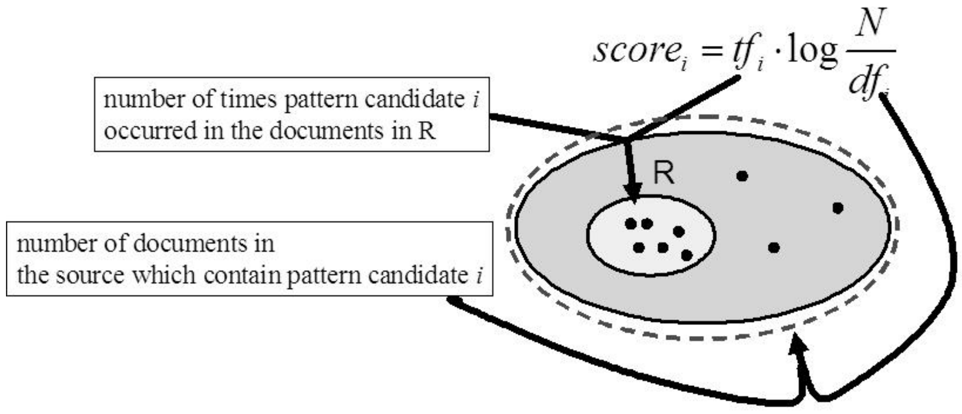

$score_i = tf_i \cdot \log \frac{N}{df_i}$
number of times pattern candidate i occurred in the documents in R
number of documents in the source which contain pattern candidate i
R

Figure 3.3: TF/IDF-based scoring function for pattern candidates

specified by the user's query. QDIE employs the cascading manner of identifying the extraction patterns. First, the system retrieves a set of documents relevant to the user's query. All the subtrees of dependency trees that conform to the definition of extraction patterns are considered as pattern candidates. The TF/IDF-based score orders the set of pattern candidates in terms of relevance to the user's query.

Chapter 4

Pattern Models

A model of extraction patterns for information extraction defines the template of extraction patterns. The template restricts how much contextual information an extraction pattern carries; for the extraction patterns to match only the relevant portions of text. Different models can contain various amount of contexts for extraction patterns. Although predicate-argument relation has been widely used for the model of extraction patterns, modeling extraction patterns is more difficult than may at first appear.

4.1 Introduction

As we have seen in the previous chapters, the central technique of information extraction is the use of extraction patterns, and automatic pattern discovery has been the focus of study [26, 40, 36]. In particular, methods have recently emerged for the discovery 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 discovery method.

In the prior work on extraction pattern discovery, the representation model of the patterns was based on a fixed set of pattern templates [27], or predicate-argument relations, such as subject-verb, and object-verb [40]. The model of our previous work [35] was based on the paths from verbal predicate nodes in dependency trees.

In this chapter, we compare the different extraction pattern representation models in relation to their ability to capture the participating entities in scenarios when used with QDIE pattern discovery framework. We present an alternative model based on arbitrary 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.

4.2 Prior models

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 [40] and the Chain model [36].

4.2.1 Predicate-Argument model

The Predicate-Argument model is based on a direct syntactic relation between a predicate and its arguments [40]. The case marking for a nominalized predicate is significantly different from the verbal predicate [1]. Thus, no simple rule can resolve finding arguments and their types for non-verbal predicates, which makes it hard to regularize the nominalized predicates automatically. Therefore, the current constraint for the Predicate-Argument model requires the root node to be a *verbal* predicate.

[1]The recent discussion of predicate-argument structure of nominalized verbs can be seen in [20, 21]

Following the definition of extraction patterns in Section 1.3, an extraction pattern in a *predicate argument model* is defined as an extraction pattern whose structure contains only a root node and its direct children. Thus, it conforms to (PRED: — (ARG1: —)(ARG2: —)...(ARGn:—)) in S-expression form. Figure 4.3(a) shows the examples obtained from a dependency tree. The root node is a verbal predicate and only its direct argument, such as a subject, an object, and its direct modifiers are included in the Predicate-Argument model.

In general, a predicate provides a simple but strong context for its arguments, which leads to good accuracy for extraction tasks. Thus, the predicate-argument model is widely used in the current IE extraction patterns.

4.2.2 Chain model

Definition

The Chain model is a representation of patterns, a path in the dependency tree passing through zero or more intermediate nodes within the tree. Thus, it conforms to (PRED: — (ARG1: — (ARG2: — (...(ARGn:—))))) in S-expression form. Figure 4.3(b) shows the examples obtained from a dependency tree. Each node contains at most one child node. Here dependency relationships are not limited to just those between a case-marked element and a predicate, but also include those between a modifier and its head element, which covers most relationships within sentences.

4.2.3 Comparison of Chain model and Predicate-Argument model

The main cause of difficulty in finding entities by extraction patterns is due to the various ways in which the participating entities appear. They appear not only as an argument of the predicate that describes the event type, but also in other places within the sentence. For example, 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.")

Figure 4.1, 4.2, 4.3[2] shows an example of an extraction task in the terror-

[2]Throughout this thesis, 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 $(t : x\ (t_1 : y_1)..(t_n : y_n))$ denotes x is the head and t

ism 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]

Benefits of Chain model

The Chain model has some advantages for pattern matching over the Predicate-Argument model in addressing the difficulty mentioned above:

- Indirect relationships

is its label, 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 [40].

[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."

The Chain model can find indirect relationships, such as the relationship between a predicate and the modifier of the argument of the predicate. For example, the pattern

"(PRED: appoint (TO: {C-POST} (OF: {C-ORG})))" can capture the relationship between "{C-ORG}" and "to be appointed" in the sentence "{C-PER} was appointed to {C-POST} of {C-ORG}."

- Relationships beyond clausal boundaries

  Chain model can capture relationships beyond clausal boundaries. The pattern "(PRED: announce (COMP: appoint (TO: {C-POST})))" can find the relationship between "{C-POST}" and "to announce". This relationship, later on, can be combined with the relationship "{C-ORG}" and "to announce" and merged into one event.

The Chain model attempts to remedy the limitations of the Predicate-Argument model. 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 [1] management succession task compared to the Predicate-Argument model.

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.

Figure 4.1: Example sentence on terrorism scenario.

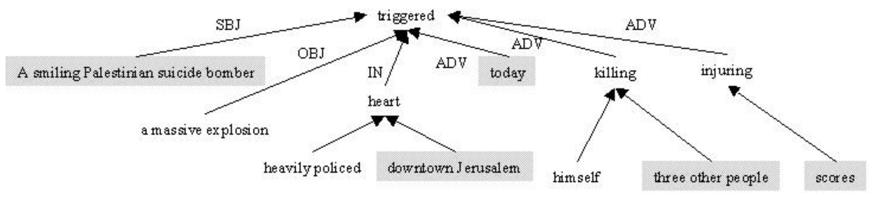


Figure 4.2: Dependency Tree of the example sentence in Figure 4.1. (The entities to be extracted are shaded in the tree).

However, the Chain model also has its own weakness in terms of accuracy due to the lack of context. For example, in Figure 4.3(a), (PRED: triggered (ADV: {C-DATE})) 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*." In the next section, we propose the novel model for extraction patterns to remedy the difficulty.

(a) Predicate-Argument
(PRED: triggered (SBJ: {C-PERSON})(OBJ: explosion)({C-DATE}-ADV))
(PRED: killing (OBJ: {C-PERSON}))
(PRED: injuring (OBJ: {C-PERSON}))

(b) Chain model
(PRED: triggered (SBJ: {C-PERSON}))
(PRED: triggered (IN: heart (OF: {C-LOCATION})))
(PRED: triggered (ADV: killing (OBJ: {C-PERSON})))
(PRED: triggered (ADV: injuring (OBJ: {C-PERSON})))
(PRED: triggered (ADV: {C-DATE}))
(PRED: injuring (OBJ: {C-PERSON}))
(PRED: killing (OBJ: {C-PERSON}))

Figure 4.3: (a) Predicate-Argument patterns and (b) Chain-model patterns that contribute to the extraction task in Figure 4.1.

4.3 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 ex-

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

Figure 4.4: Subtree model patterns that contribute the extraction task on the example sentence in Figure 4.1.

traction pattern candidate. As shown in Figure 4.4, 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 (PRED: triggered (OBJ: explosion)(ADV: {C-DATE})).

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 increases. 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 (PRED: shoot (OBJ: {C-PERSON})(TO: death)) and B is (PRED: shoot (OBJ: {C-PERSON}))), 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.

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 [31], further filtering was done by requiring a pattern candidate to contain at least one Named Entity class.

In the following sub-sections, we discuss an automatic procedure to learn extraction patterns using the QDIE framework. The general description of the three stages, *preprocessing*, *document retrieval*, and *pattern extraction*, can be seen in Section 3.2. Here we focus on the detail of the *pattern extraction* stage; in particular, the ranking function and parameter tuning.

4.3.1 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 [3] 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. It efficiently avoids the construction of duplicate patterns by keeping track of their occurrence in the corpus, 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

$$score_i = tf_i \cdot \left(\log \frac{N}{df_i}\right)^{\beta} \tag{4.1}$$

where tf_i is the number of times that subtree i appears across the documents in the *relevant document set*, 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.

4.3.2 Parameter Tuning for Ranking Function

The β in Equation 4.1 is used to parameterize the weight on the IDF portion of the ranking function. As we pointed out in Section 4.3, 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. For example, high β value prefers (PRED: triggered (OBJ: explosion)(ADV: {C-DATE})) to (PRED: triggered (ADV: {C-DATE})) in the terrorism scenario. Figure 4.5 shows the improvement of the extraction performance by tuning β on the entity extraction task.

For unsupervised tuning of β, we used a pseudo-extraction task, instead of using held-out data in case of 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

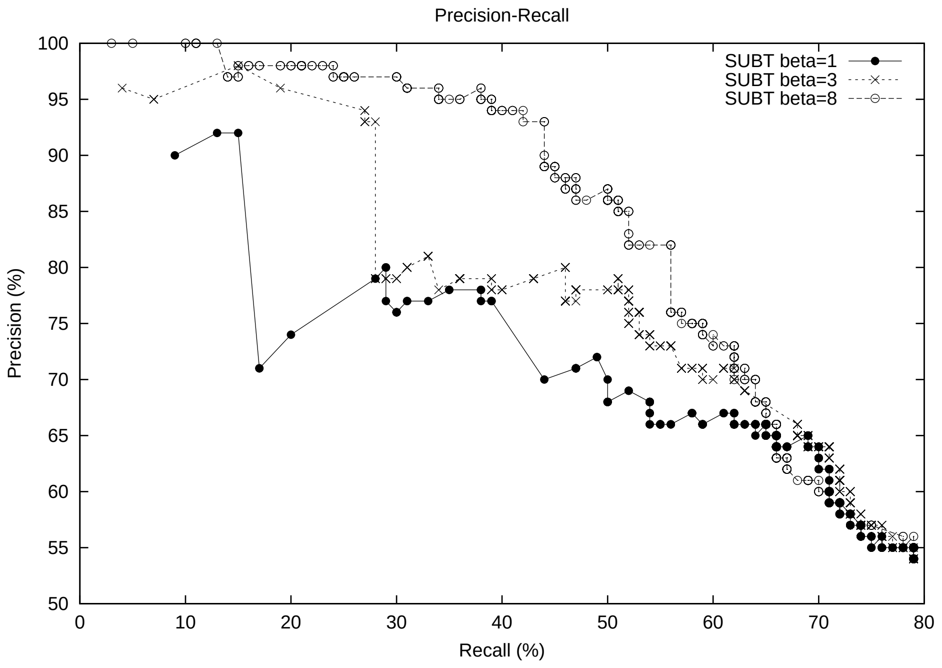


Figure 4.5: Comparison of Extraction Performance with Different β

over the entire range of recall for the text classification task is maximized [5].

The document set for text classification is composed of documents retrieved by the same IR system as used for pattern discovery, plus the same number of documents picked up randomly. Note that 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

[5]The optimum shape of the precision-recall curve given a set of extraction patterns can be obtained by ranking all the relevant patterns higher than the irrelevant patterns, which makes the maximum *area* of the precision-recall curve.

of the actual extraction performance shows that this tuning has high correlation with the extraction performance (Spearman correlation coefficient $r_s = 0.83$).

4.4 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)[6], and the Chain model (CH) in [36].

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

[6]This is a restricted version of [40] constrained to have a single place-holder for each pattern, while [40] 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)	See Table 4.2(a).	See Table 4.2(b).
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 4.1: Task Description and Statistics of Test Data

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 [1], 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.

(a) 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.

(b) Arrest: 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.

Table 4.2: IR Description of Test Data

The source document set from which the extraction patterns are learned consists of 117,109 Japanese Mainichi Newspaper articles from 1995. All the sentences are morphologically analyzed by JUMAN [16] and converted into dependency trees by KNP [17]. Regardless of the model of extraction patterns, the pattern discovery follows the procedure described in Section 3.2. 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 *Sus-*

pect, Arresting Agency, Charge (slots) correspond to C-PERSON, C-ORG, C-OFFENCE (NE-classes), respectively, in the Arrest scenario. [7]

For each model, we get a list of the pattern candidates ordered by the ranking function discussed in Section 4.3.1 after filtering. The result of the performance is shown (Figure 4.6) 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 4.1 for detail.)

Figure 4.6(a,b) shows the performance on the entity extraction task on both domains by plotting top-n relevant extraction patterns in precision and recall. In both graphs, SUBT, CH, and PA corresponds to Subtree, Chain and Predicate-Argument model, respectively.

Figure 4.6(a) shows the precision-recall curve of extraction patterns for each model on the Succession Scenario. At lower recall levels (up to 35%),

[7]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.

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 4.6(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 4.6(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.

4.5 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 of appointing". 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.

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 4.6 (a), the large drop at the recall level 55%

for Subtree model and at 48% for Chain model 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 4.3.1 is the same for both models, the Subtree model can also produce the patterns that are likely extract relevant entities (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.

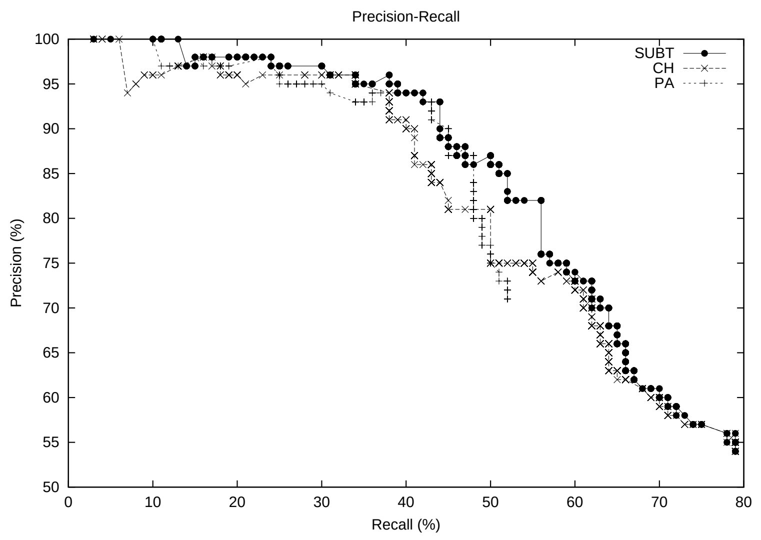
Without taking case marking for nominalized predicates into account, the Predicate-Argument model excludes some highly contributing patterns with nominalized predicates, as some example patterns show in Figure 4.7 and Figure 4.8. Also, chains of modifiers could be extracted only by the Subtree and Chain 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 4.7, 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.

4.6 Conclusion

In this chapter, we explored alternative models for the automatic discovery 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.

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

(a)


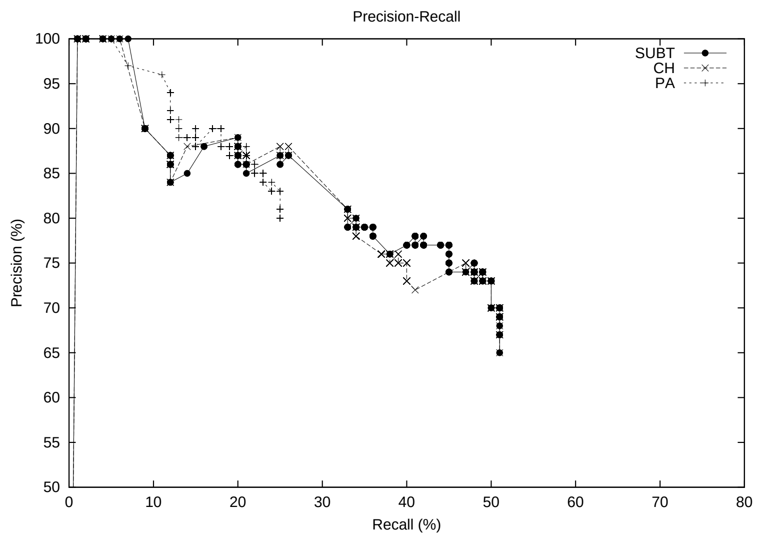
(b)


Figure 4.6: Extraction Performance: (a) Succession Scenario ($\beta = 8$), (b) Arrest Scenario ($\beta = 4$)

Pattern	Cor	Inc	SUBT	Chain	PA
(({C-PERSON}{C-POST}-OF) {C-PERSON} *shoukaku*) *"promotion of {C-POST}{C-PERSON}"*[8]	26	1	yes	yes	no
(((*daihyo-ken*-SBJ) *aru*-REL) {C-POST}) *"{C-POST} with ministerial authority"*	4	0	yes	yes	no
((((*daihyo-ken*-(*no*)SBJ) *aru*-REL) {C-POST}-TO) *shunin-suru*) *"be appointed to {C-POST} with ministerial authority"*	2	0	yes	yes	no
(({C-ORG}-SBJ) *happyo-suru*) *"{C-ORG} reports"*	16	3	yes	yes	yes
(({C-ORG}-SBJ) (*jinji*-OBJ) *happyo-suru*) *"{C-ORG} report personnel affair"*	4	0	yes	no	yes

Figure 4.7: Examples of extraction patterns and their contribution in Management Succession scenario: For each pattern, the table shows its English translation, the number of correct matches (Cor), the number of incorrect matches (Incorrect) and whether it conforms with Subtree (SUBT), Chain (Chain) and/or Predicate-Argument (PA) model.

Pattern	Cor	Inc	SUBT	Chain	PA
(({C-PERSON}{C-POST}-APPOS) {C-NUM})	54	22	yes	yes	no
((({C-PERSON}{C-POST}-APPOS) {C-NUM}-OBJ) *taiho-suru*) *"arrest {C-PERSON}{C-POST}, {C-NUM}"*	17	1	yes	yes	no
(((*mushoku*-APPOS) {C-PERSON}{C-POST}-APPOS) {C-NUM}) *"{C-PERSON}{C-POST}, {C-NUM}, unemployed"*	11	0	yes	yes	no

Figure 4.8: Examples of extraction patterns and their contribution in Arrest scenario: For each pattern, the table shows its English translation, the number of correct matches (Cor), the number of incorrect matches (Incorrect) and whether it conforms with Subtree (SUBT), Chain (Chain) and/or Predicate-Argument (PA) model.

Chapter 5

Case Study: Crosslingual Information Extraction

The most straightforward way to exploit the QDIE framework presented in Chapter 3 is to build an IE system that customizes itself to the new task specified by the user's query. As a case study of the QDIE framework, this chapter presents an experimental self-customizing IE system with an additional dimension of portability – IE from a source text in a different language than the target language.

5.1 Introduction

As we discussed in previous chapters, the QDIE system provides portability in domain/task by allowing the user to specify the task. The model of extraction patterns the QDIE system uses is designed to be general enough for the systems in different languages; only Named Entities and dependency parser are the necessary source-language-specific linguistic analysis tools. In this case study, we addressed two central aspect of portability for information extraction — across domains and languages.

The first half of this chapter discusses the experimental information extraction system which takes Japanese source data and generates English tables of extracted information, and that can be easily adapted to new extraction tasks. We want to minimize the human intervention required for customization to a new scenario (type of facts or events of interest), and allow the user to interact with the system entirely in English. As a prototype, we have developed the pre-CODIE system, an experimental (pre-) *C*ross-lingual *O*n-*D*emand *I*nformation *E*xtraction system that extracts facts or events of interest from Japanese source text without requiring user knowledge of Japanese.

The second half discusses the role of machine translation in cross-lingual information extraction. We compared two approaches for cross-lingual information extraction; one that translates the entire source text and runs the QDIE system on the translated text to find extraction patterns and does extraction, and the one that translates the user's query into the language of the source text, henceforth *source language*, and runs the QDIE system in the source language.

5.2 pre-CODIE System

5.2.1 Overview

Incorporating the QDIE system as a module for pattern discovery, the extraction functionality of pre-CODIE system is driven by the query from the user. The user starts the procedure by specifying the type of facts or events of interest in the form of a narrative description, and then pre-CODIE customizes itself to the topic by acquiring extraction patterns based on the user's description. Pre-CODIE, as an early attempt at a fully-automated system, still needs user interaction for template definition (adding/deleting the slots in the template) and slot assignment (assigning a pattern to one of the slots in the template); automating these steps is left as future work. (For more discussion of fully-automated ODIE system, see Chapter 6.)

Pre-CODIE interacts with its user entirely in English; even for slot assignment of the extraction patterns, the system translates the Japanese extraction patterns, which are based on subtrees of a dependency tree as discussed in Chapter 4, by word-to-word translation of each lexical item in the patterns. For ease of use, the Japanese extraction patterns are not only translated into English, but also shown with translated example sentences which match the pattern.

5.2.2 System Architecture

Pre-CODIE is implemented as an integration of several modules, as shown in Figure 5.1: translation, information retrieval, pattern discovery, and extraction.

First, the system takes the narrative description of the scenario of interest as an English query, and the Translation module (off-the-shelf IBM *Internet King of Translation Bilingual* [13] [1] system) generates a Japanese query.

The QDIE framework is used as an IR module and Pattern Discovery module. The IR module retrieves a set of *relevant documents* from Japanese

[1] At the time of implementation, we used Version 4 of the system.

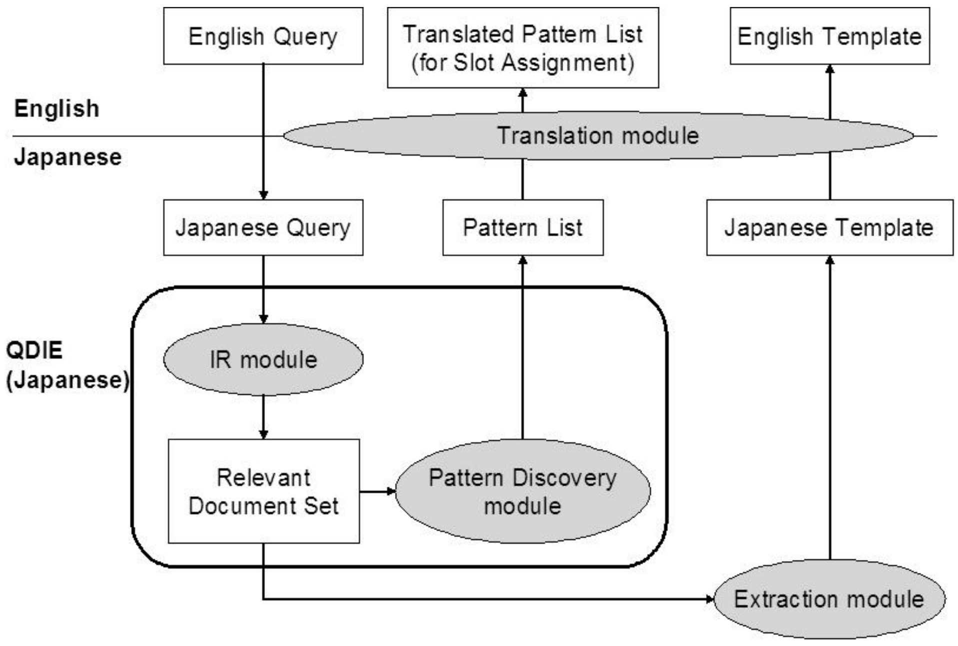


Figure 5.1: pre-CODIE System Architecture

Mainichi Newspaper from 1995. Then, the Pattern Discovery module produces a list of extraction patterns from the *relevant document set* sorted by their relevance to the scenario. (See Chapter 3 for the detailed description.)

Pre-CODIE system asks the user to assign each Named Entity class name in the patterns to one of the slots in the template, which the user can freely customize in the Slot Configuration stage. Finally, the Extraction module performs the pattern matching with slot-assigned patterns to each text in the *relevant document set* and generates a filled Japanese template, which is translated slot-by-slot into English for the user.

5.2.3 An Example procedure: Management Succession

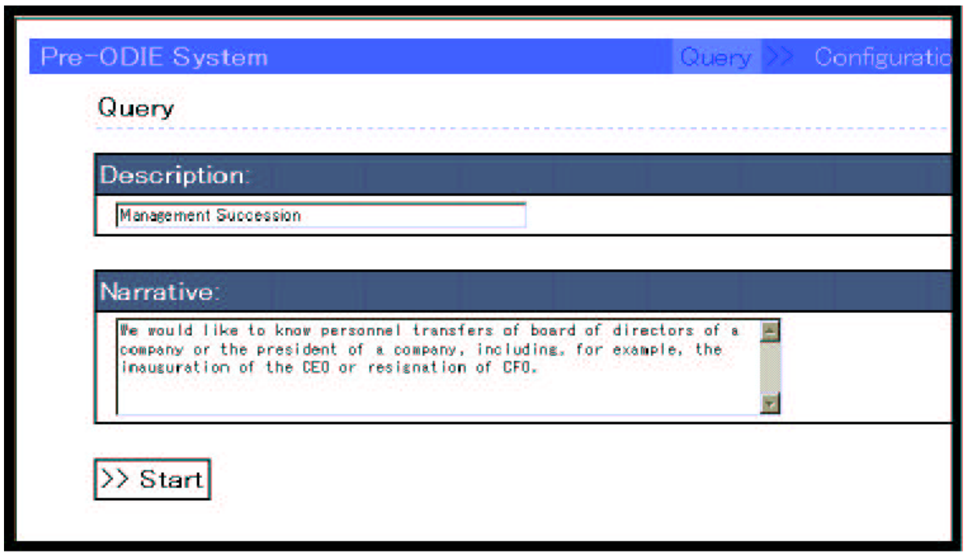
From the user's point of view, pre-CODIE works as follows with the screen shots in Figure 5.2– 5.5.

1. **Query:** The user types in the narrative description of the scenario of interest, one phrase in "description" text-box and more detail optionally given in "narrative" text-box. For the example shown in Figure 5.2, a narrative description is given as follows, *"Management Succession: We would like to know personnel transfers of board of directors of a company or the president of a company, including, for example, the inauguration of the CEO or resignation of CFO."*.

2. **Slot Configuration:** The user adds and/or deletes the slots in the template. In the example shown in Figure 5.3, the user added *"Person-In"*, *"Person-Out"*, *"Post-In"*, *"Post-Out"*, and *"Organization"*.

   The system then performs pattern discovery and presents patterns.

3. **Slot Assignment:** The user assigns a slot to each placeholder of each pattern by choosing one of the slots defined in step 2. During this stage,



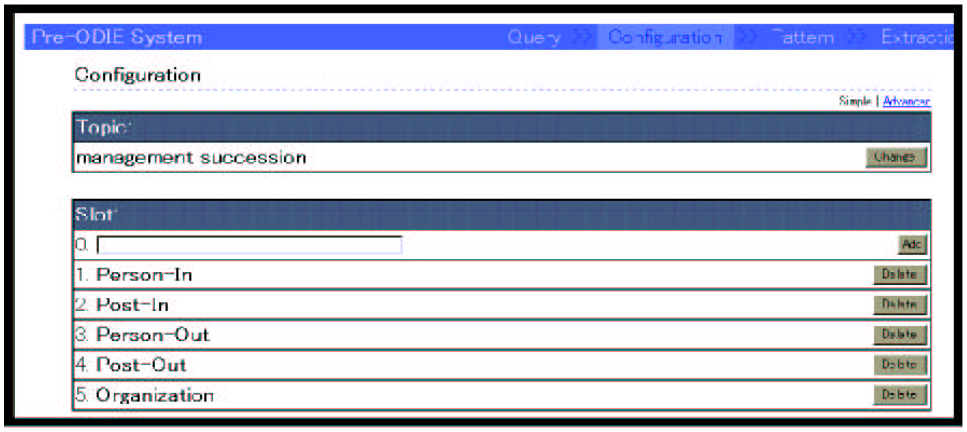
Description:

Management Succession

Narrative:

We would like to know personnel transfers of board of directors of a company or the president of a company, including, for example, the inauguration of the CEO or resignation of CFO.

Figure 5.2: Step 1: Query: The user provides the scenario description in the Description (a set of keywords) and/or Narrative (a narrative description with a few sentences).



Configuration

Topic:

management succession

Slot:

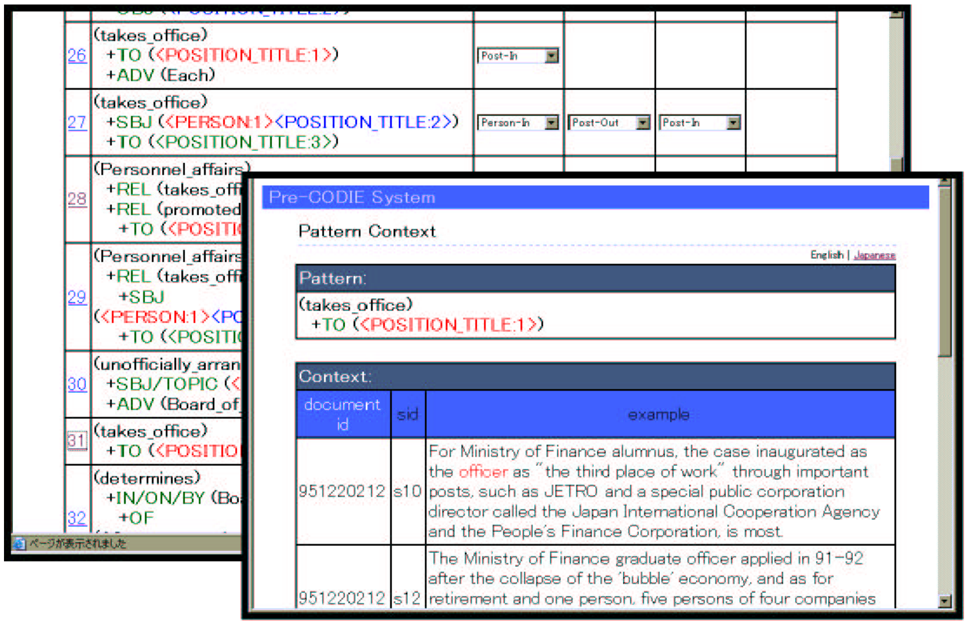
0.

1. Person-In
2. Post-In
3. Person-Out
4. Post-Out
5. Organization

Figure 5.3: Step 2: Slot Configuration: The user adds and/or deletes the slots in the template freely.

the user is allowed to see the example of translated sentences with the match of the pattern highlighted. This will make it easier for the user to understand what each pattern aims to extract. In Figure 5.4, the user tries to assign "*(PRED: takes_office (TO: {POSITION_TITLE}))*" to *"Post-In"* while observing the examples where the pattern appears.

4. **Extraction:** The user gets the extracted template and repeats this procedure until the user gets the right template by going back to step 3 to change and/or add slot assignments, and by going back to step 2 to delete and/or add slots in the template.



26	(takes_office) +TO (<POSITION_TITLE:1>) +ADV (Each)	Post-In			
27	(takes_office) +SBJ (<PERSON:1><POSITION_TITLE:2>) +TO (<POSITION_TITLE:3>)	Person-In	Post-Out	Post-In	
28	(Personnel_affairs) +REL (takes_offi +REL (promoted +TO (<POSITI				
29	(Personnel_affairs +REL (takes_offi +SBJ (<PERSON:1><PO +TO (<POSITI				
30	(unofficially_arran +SBJ/TOPIC (< +ADV (Board_of				
31	(takes_office) +TO (<POSITIO				
32	(determines) +IN/ON/BY (Bo +OF				

ページが表示されました

Pre-CODIE System

Pattern Context

English | Japanese

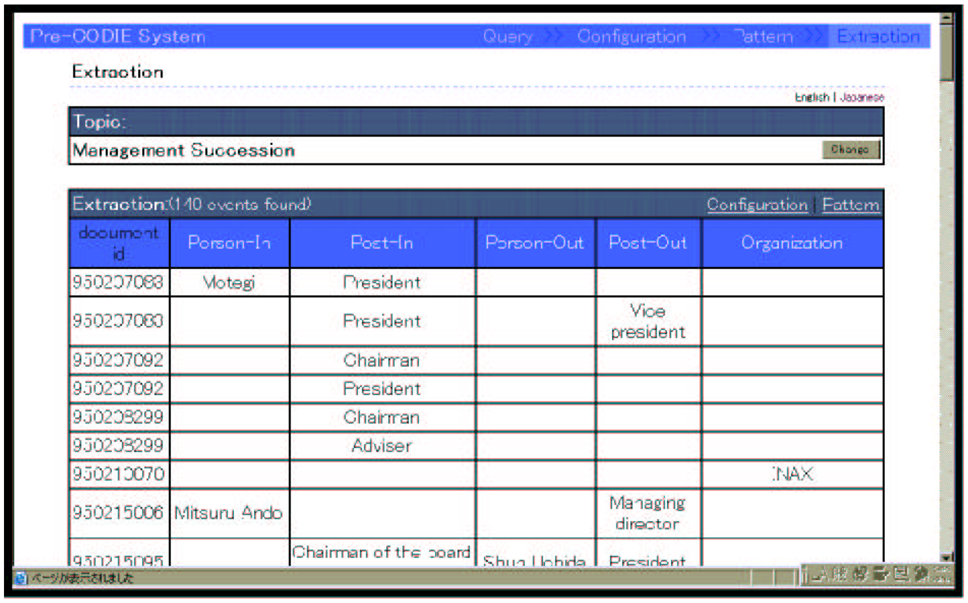
Pattern:

```
(takes_office)
  +TO (<POSITION_TITLE:1>)
```

Context:

document id	sid	example
951220212	s10	For Ministry of Finance alumnus, the case inaugurated as the officer as "the third place of work" through important posts, such as JETRO and a special public corporation director called the Japan International Cooperation Agency and the People's Finance Corporation, is most.
951220212	s12	The Ministry of Finance graduate officer applied in 91-92 after the collapse of the 'bubble' economy, and as for retirement and one person, five persons of four companies

Figure 5.4: Step 3: Slot Assignment: The user assigns a slot to each placeholder of the pattern by choosing one of the slots defined in step 2. During this stage, the example of translated sentences can be seen.



Extraction

English | Japanese

Topic:

Management Succession

Extraction:(140 events found) Configuration | Pattern

document id	Person-In	Post-In	Person-Out	Post-Out	Organization
950207083	Motegi	President			
950207083		President		Vice president	
950207092		Chairman			
950207092		President			
950208299		Chairman			
950208299		Adviser			
950210070					INAX
950215006	Mitsuru Ando			Managing director	
950215095		Chairman of the board	Shuji Uchida	President	

Figure 5.5: Step 4: Extraction: The user can go back step 2 or 3 and repeat until he/she gets the desirable result.

Thus, pre-CODIE system allows the user to specify the event of interest and the template to fill in. We observed that pre-CODIE system works in various scenarios, including prime-rate fluctuation (Date, Rate), president election (Candidate, Post, Political_Party, Date), and bank failure (Organization, Location, Date, Deficit).

In the next section, we will discuss how well pre-CODIE system works by comparing it to the alternative approach for cross-lingual information extraction, reported in Sudo et al. [37]. We also discuss the role of machine translation in cross-lingual information extraction.

5.3 Evaluation of Cross-lingual IE System and Role of Machine Translation

In this section, we evaluate the QDIE system with the alternative method for efficient automatic pattern discovery for the CLIE (Cross-Lingual IE) system. The alternative method translates the whole source text into the target language first, and runs the pattern discovery procedure on the text in the target language.

5.3.1 Cross-lingual Information Extraction

Riloff et al. 2002 [29] present several approaches to cross-lingual information extraction (CLIE). They describe the use of ”cross-language projection” for CLIE, exploiting the word alignment of documents in one language and the same documents translated into a different language by a machine translation (MT) system. They conducted experiments between two relatively close languages, English and French. In the experiment reported here, we will explore CLIE for two more disparate languages, English and Japanese.

The QDIE system can be used in a cross-lingual setting, and thus, the resulting cross-lingual version of the QDIE system can minimize the requirement of knowing the language of the source text (source language) from the user. Figure 5.6 shows two possible ways to achieve this goal.

1. **Translation-based QDIE System:** This is realized by translating all the documents of the source language (Japanese) into the target language (English), then running the monolingual version of the QDIE system for the target language (English). (Figure 5.6(A)) This way of implementation only requires a machine translation system to translate the source text and does not require the linguistic analysis tools for pattern discov-

ery, such as a NE tagger and dependency parser, in the source language. However, the pattern discovery needs to be done on the translated documents which may contain translation errors.

2. **Cross-lingual QDIE System:** On the other hand, as shown in Figure 5.6(B), one can first translate the scenario description into the source language (Japanese) and use it for the monolingual QDIE system for the source language (Japanese), assuming that we have access to the tools for pattern discovery; namely, a Named Entity tagger and dependency parser. Each item in the extracted table is translated into the target language (English). pre-CODIE system, discussed in Section 5.2, was implemented in this way.

As we shall demonstrate, if basic linguistic analysis tools are available for the source language, it is possible to achieve a better CLIE performance by learning patterns in the source language. In the next section, we assess the difficulty of learning extraction patterns from the translated source language document set caused by the errors of the MT system and/or the differences of grammatical structure of the translated sentences. We address specifically:

1. The accuracy of NE tagging on MT-ed source documents and the use of

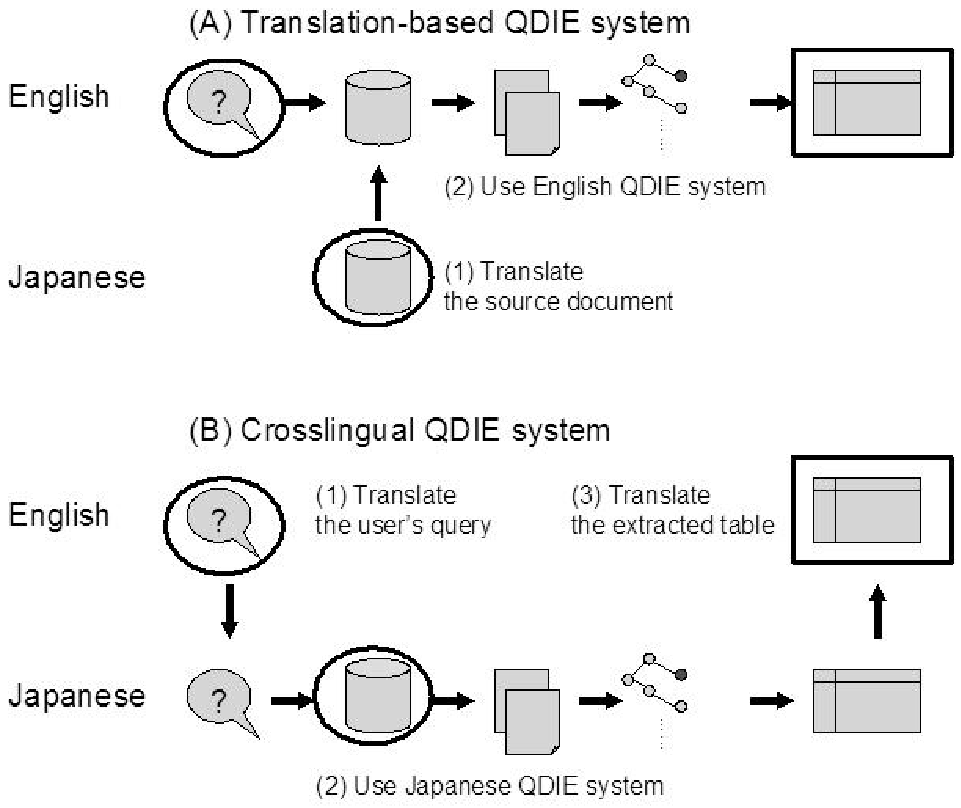


Figure 5.6: Translation-based QDIE System(A) vs Cross-lingual QDIE System(B): The user's query (English), the source document (Japanese) and the target extracted table (English) are highlighted.

cross-language projection.

2. How the structural difference in source and target language affects the extracted patterns.

3. The reduced frequency of the extracted patterns, which makes it difficult for any measurement of pattern relevance to distinguish the effective patterns of low frequency from the noise patterns.

5.3.2 Experiments

To evaluate the relevance of extraction patterns automatically learned for CLIE, we conducted experiments for the Translation-based QDIE system and the Cross-lingual QDIE system on the *entity extraction task*, which is to identify all the entities participating in relevant events in a given set of Japanese texts.

5.3.2.1 Experimental Setting

Since general NE taggers either are trained on English sentences or use manually created rules for English sentences, the deterioration of the NE tagger's performance cannot be avoided if it is applied to the MT-ed English sentences.

This causes the Translation-based QDIE system to identify fewer pattern candidates from the *relevant documents* since a pattern candidate must contain at least one of the NE types.

To remedy this problem, we incorporated "cross-language projection" (Riloff et al. 2002 [29]) only for Named Entities. We used word alignment obtained by using Giza++ [24] to get names in the English translation from names in the original Japanese sentences. Note that it is extremely difficult to make an alignment of the right dependencies where one language explicitly renders a marker as a word and the other does not. So, direct application of [29] is not suitable for this experiment.

We compare the following three systems in this experiment.

1. Cross-lingual QDIE system

2. Translation-based QDIE system with word alignment

3. Translation-based QDIE system without word alignment

5.3.2.2 Data

The scenario for this experiment is the Management Succession scenario of MUC-6[1], where corporate managers assumed and/or left their posts. We used a much simpler template structure than the one used in MUC-6, with Person, Organization, and Post slots. To assess system performance, we measure the accuracy of the system at identifying the participating entities in a management succession event. This task does not involve grouping entities associated with the same event into a single template, in order to avoid possible effects of merging failure on extraction performance for entities.

The source document set from which the extraction patterns are learned consists of 132,996 Yomiuri Newspaper articles from 1998. For our Cross-lingual QDIE system, all the documents are morphologically analyzed by JUMAN [16] and converted into dependency trees by KNP [17]. For the Translation-based QDIE system, all the documents are translated into English by a commercial machine translation system (IBM "King of Translation"), and converted into dependency trees by a corpus-based dependency parser. We retrieved 1500 documents as *relevant documents*.

We accumulated the test set of documents by a simple keyword search. The test set consists of 100 Yomiuri Newspaper articles from 1999, out of which only 61 articles contain at least one management succession event. Note that all NE in the test documents both in the original Japanese and in the translated English sentences were identified manually. We tried to avoid the difference in

Documents (relevant + irrelevant)	100 (61 + 39)
Names (relevant + irrelevant)	Person: 173 + 651 Org: 111 + 709 Post: 210 + 626

Table 5.1: Statistics of Test Data

NE tagger's performance on each language, 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. Table 5.1 shows the detail of the test data.

5.3.2.3 Results

Each pattern acquisition system outputs a list of the pattern candidates ordered by the ranking function. The resulting performance is shown as a precision-recall graph for each subset of top-n ranked patterns where n ranges from 1 to the number of pattern candidates. The parameters for each system are tuned to maximize the performance on a separate validation data.

The association of NE classes in the matched patterns and slots in the tem-

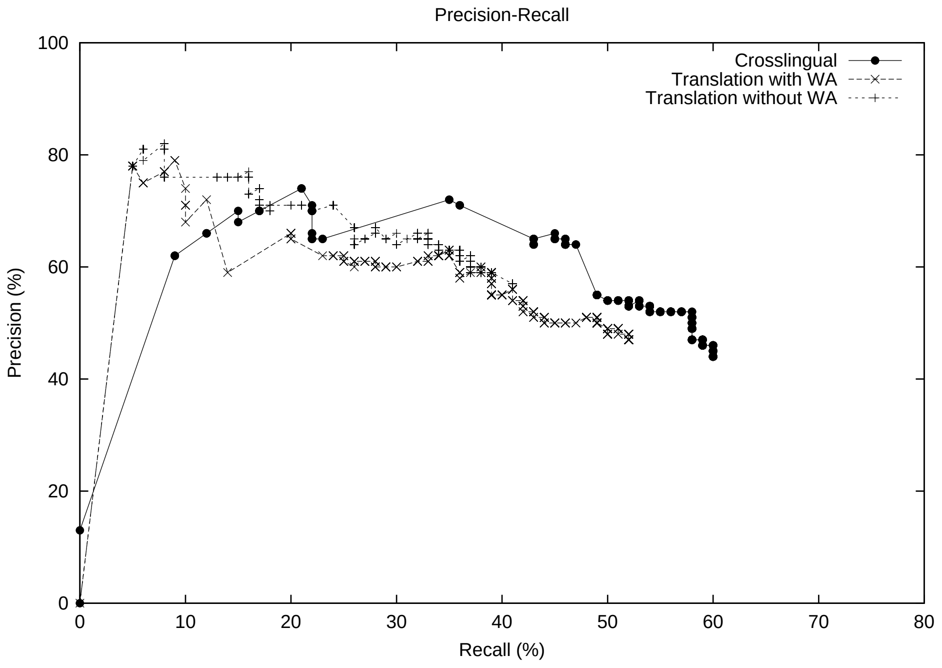


Figure 5.7: Performance Comparison on Entity Extraction Task

plate is made automatically; *Person, Organization, Post* (slots) correspond to C-PERSON, C-ORG, C-POST (NE-classes), respectively, in the Management Succession scenario.

Figure 5.7 shows the precision-recall curve for the top 1000 patterns acquired by each system on the entity extraction task. Cross-lingual QDIE system reaches a maximum recall of 60%, which is significantly better than Translation-based QDIE with word alignment (52%) and Translation-based

QDIE without word alignment (41%). Within the high recall range, Crosslingual QDIE system generally had better precision at the same recall than Translation-based QDIE systems. At the low recall range ($< 20\%$), the performance is rather noisy.

Translation-based QDIE without word alignment performs similarly to Translation-based QDIE with word alignment up to its maximum recall (41%). Translation-based QDIE with word alignment reached 10% higher maximum recall (52%).

5.3.3 Problems in Translation

The detailed analysis of the result revealed the effect of several problems caused by the MT system. The current off-the-shelf MT system's output resulted in difficulty in using it as a source of extraction patterns. In this section we will discuss the types of differences between the source and target languages, and their effect on pattern discovery.

Lexical differences Abbreviations in the source language may not have their corresponding short form in the target language. For example, "Kei-Dan-Ren" is an abbreviation of "*Kei*zai *Dan*tai *Ren*go-kai" which is an organization whose English translation is "Japan Federation of Economic Organizations". Such abbreviations may not be listed in the dictionary of the MT system. In such cases, the literal translation of the abbreviation may be difficult to recognize as a name and is likely to be treated as a common noun phrase.

Structural differences Some phrases in the source language may have more than one relevant translation. Depending upon the context where a phrase appears, the MT system has to choose one among the possible translations. Moreover, the MT system may make a mistake, of course, and output an erroneous translation. This results in a diverse distribution of extraction patterns in the target language. Figure 5.8 shows an example of such a case. Suppose an extraction pattern (({C-POST}-*ni*) *shuninsuru*) appears 20 times in the original Japanese document set, out of which it may be translated 10 times as (be appointed (to ({C-POST}))), 5 times as (assume ({C-POST})), 3 times as (be inaugurated (as ({C-POST}))), and 2 times as an erroneous translation. Some of the lower frequency translated patterns will be ranked lower by the scoring function and so will be hard to distinguish from noise.

Figure 5.9 shows an example of the case where the context around the name did not seem to be translated properly, so the dependency tree for the

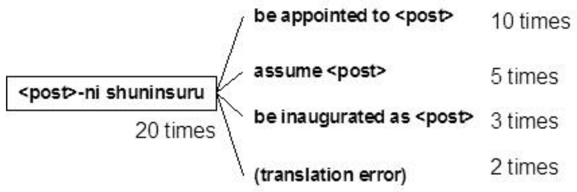
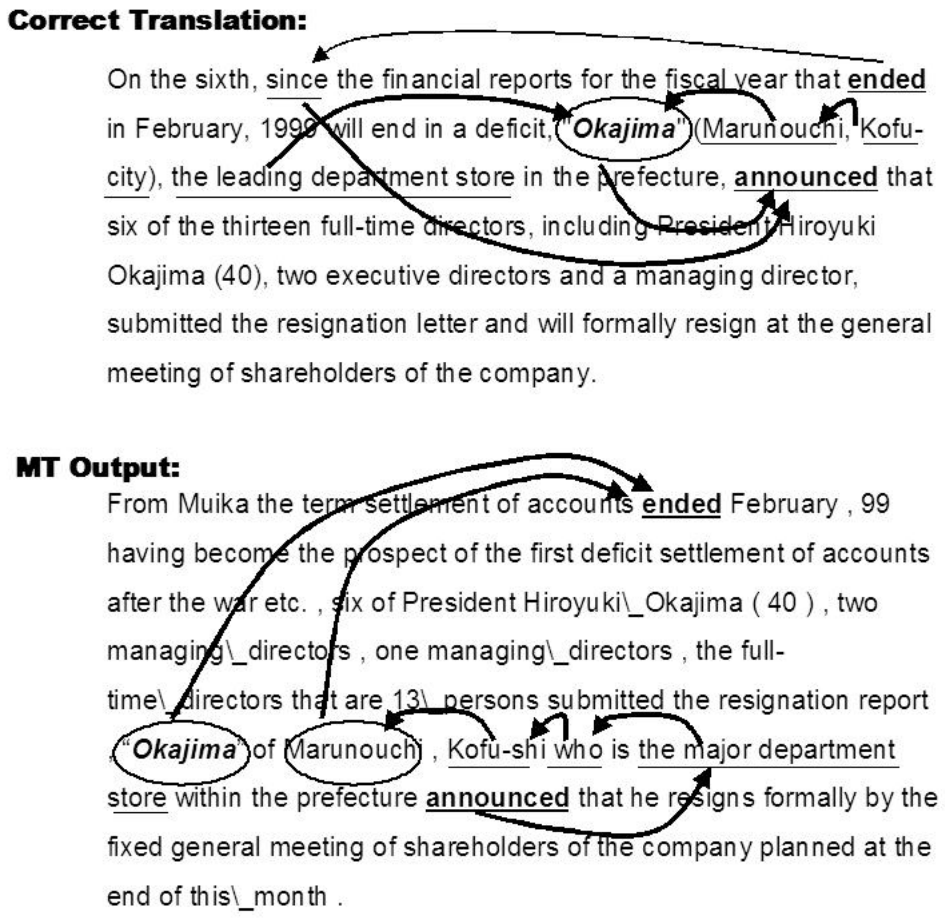


Figure 5.8: Example of Structural Difference in Translation: The translation of a Japanese expression into several English different expressions including erroneous ones.

sentence was not correct. The right translation is "Okajima announced that President Hiroyuki Okajima, 40 years old, resigned formally ..." which results in the dependency between the main verb "announce" and the company "Okajima". The translation shown in Figure 5.9 not only shows incorrect word-translations, but also shows ungrammatical structure, including too many relative clauses. The structural error causes the errors in the dependency parse tree including having "end" as a root of the entire tree and the wrong dependency from "announced" to "the major department" in Figure 5.9 [2]. Thus, the accumulation of the errors resulted in missing the organization name "Okajima".

[2]The head is "the major department" and "announced" is modifying the head.



Correct Translation:

On the sixth, since the financial reports for the fiscal year that **ended** in February, 1999 will end in a deficit, "***Okajima***" (Marunouchi, Kofu-city), the leading department store in the prefecture, **announced** that six of the thirteen full-time directors, including President Hiroyuki Okajima (40), two executive directors and a managing director, submitted the resignation letter and will formally resign at the general meeting of shareholders of the company.

MT Output:

From Muika the term settlement of accounts **ended** February , 99 having become the prospect of the first deficit settlement of accounts after the war etc. , six of President Hiroyuki_Okajima (40) , two managing_directors , one managing_directors , the full-time_directors that are 13_persons submitted the resignation report , "***Okajima***" of Marunouchi , Kofu-shi who is the major department store within the prefecture **announced** that he resigns formally by the fixed general meeting of shareholders of the company planned at the end of this_month .

Figure 5.9: Example of Translation Errors: Arrows indicate the correct dependencies in Correct Translation, and the dependency parser's output of the sentence in MT output, respectively. In MT output, the structure error causes the organization "Okajima" depend to the verb, *"end"*, erroneously analyzed as a main predicate.

Out-of-Vocabulary Words The MT system may not have a word in the source language dictionary, in which case some MT systems output it in the original script in the source language. This happens not only for names but also for sentences which are erroneously segmented into words. Such problems, of course, may make it hard to detect Named Entities and get a correct dependency tree of the sentence.

However, translation of names is easier than translation of contexts; the MT system can output the transliteration of an unknown word. In fact, name translation of the MT system we used for this experiment is better than the sentence translation of the same MT system. The names appropriately extracted from Japanese documents by the Cross-lingual QDIE system, in most cases, are correctly translated or transliterated if no equivalent translation exists.

5.3.4 Related Work

The work closest to ours is [29]. They showed how IE learning tools, bitext alignment, and an MT system can be combined to create CLIE systems between English and French. They evaluated a variety of methods, including one similar to our Translation-based QDIE. Their approaches were less reliant on language tools for the “source” language (in their case, French) than our Cross-lingual-QDIE system. On the other hand, their tests were made on a closer language pair (English - French). We expect that the performance gap between Translation-based IE and Cross-lingual IE is more pronounced with a more divergent language pair like Japanese and English.

There are interesting parallels between our work and that of [10], who discussed the role of machine translation in a cross-lingual summarization system which produces an English summary from Arabic text. Their system took the same path as our Cross-lingual QDIE: summarizing the Arabic text directly and only translating the summary, rather than translating the entire Arabic text and summarizing the translation. They had similar motivations: different translations produced by the MT system for the same word in different contexts, as well as translation errors, would interfere with the summarization process.

The trade-offs, however, are not the same for the two applications. For summarization either path requires an MT system which can translate entire sentences (either the original text or the summary). Translation-based QDIE has a similar requirement, but Cross-lingual QDIE reduces the demands on MT: only query translation and name translation are required.

5.4 Conclusion and Future Work

We have described the experimental pre-CODIE system. With some necessary human intervention for slot-assignment, pre-CODIE system can automatically identify a set of extraction patterns and create filled-out templates for the events specified by the user's query.

We discussed the difficulty in cross-lingual information extraction caused by the translation of the source documents using an MT system. The experimental result for entity extraction suggests that exploiting some basic tools available for the source language will boost the performance of the whole CLIE system.

We intend to investigate whether further performance gain may be obtained by introducing additional techniques for query translation. These techniques, including query translation on expanded queries and building a translation dictionary from parallel corpora, are currently used in cross-lingual information retrieval [18].

Chapter 6

Future Work

In the previous chapters, we introduced the unsupervised pattern discovery, QDIE, framework from various aspects. We have shown that it is possible to make an automatic pattern discovery procedure combining document retrieval and pattern scoring using a TF/IDF-based function. We have also discussed how further improvement in extraction performance may be achieved by choosing a better extraction pattern model. The portability of the QDIE framework across languages has been shown by giving an example of a cross-lingual IE system, pre-CODIE system. However, there is no reason we should stop seeking further improvement and potential applications. In this chapter, we will discuss three of the prospects of the QDIE framework; namely, the improvement in recall by pattern expansion, the improvement in precision by relevance feedback using discovered extraction patterns and the application to on-demand information extraction.

Pattern Expansion

An implicit assumption on pattern discovery, including the QDIE framework, is that a large number of documents cover most of the extraction pattern variations in a given domain. In practice, however, there are various kinds of difficulties which prevent the discovered set of extraction patterns from a particular source text from covering most of the events in a new text. In this section, we report our preliminary study on an upper-bound in the coverage of discovered extraction patterns from a newspaper corpus for one year and propose a method to find new extraction patterns that do not appear in the source text.

Performance of NE tagger: Since our definition of extraction patterns uses Named Entities as a matching variable, the accuracy of the NE tagger limits the upper bound of any IE tasks. In our preliminary study on MUC-6 training data, only 83% of Person, Organization and Post instances are correctly covered by our NE tagger [1]. Since the instances that cannot be recognized by the NE tagger will never be matched by any extraction patterns, this shows the limit of extraction based on NE instances.

Pattern Expansion by Word Replacement The set of extraction patterns is expanded by replacing a word in the context of a pattern with another word that is semantically equivalent to the original word. Figure 6.1 shows how a new pattern, (PRED: report (SBJ: {C-ORG})), can be obtained by replacing a verb, *announce*, with the verb, *report*, in an existing pattern (PRED: announce (SBJ: {C-ORG})). Note that the verb "*report*" may only appear with a different subject and that there is no occurrence of the pattern, (PRED: report (SBJ: {C-ORG})), in the source document.

In the following items, we discuss the expansion of 1000 patterns which are discovered using the QDIE framework (Chapter 3 and the subtree model discussed in Section 4.3) applied to 1500 retrieved paragraphs from the New York Times 1995.

We conducted a preliminary study of pattern expansion by replacing a word in an extraction pattern by its semantic equivalents. The new pattern set was

[1]In the following preliminary study, we used Sekine and Nobata's NE tagger [30] and used only person, organization, location, position_title, date, time, and percent classes.

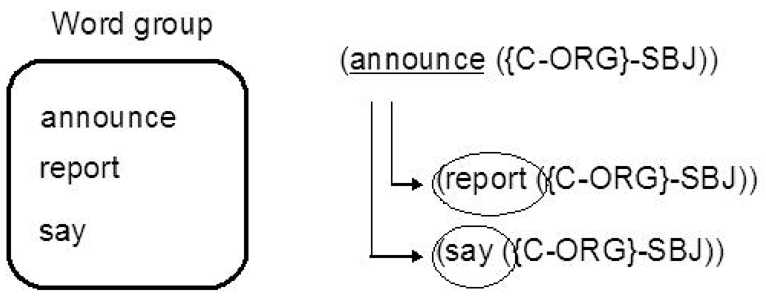


Figure 6.1: Example of Pattern Expansion by word replacement: The new patterns are obtained by replacing *announce* in the existing patterns with the other words, *repeat* and *say*, in the same word group.

obtained by adding new patterns that are created by replacing a word in an existing pattern with the semantically equivalent word. The word candidates for replacement are found by using WordNet and word clustering.

- **WordNet:** For each word in the pattern, create a cluster whose members are the hypernym (one-level up) and hyponyms (all descendants in the tree) of the target word using WordNet.

- **Clustering:** A set of words is created from all the words in the retrieved documents. Word clustering is used to measure contextual proximity of words. We used the CLUTO package [41, 42] and used the parent and children nodes in the dependency tree, and bag-of-words within two

	Max Recall
Baseline (1000 patterns)	54
WordNet	68
WordNet + Clustering	69
All NEs	83

Table 6.1: Maximum recalls in entity extraction task

words as features for word clustering.

Table 6.1 shows the maximum recall on the entity extraction task with the expanded pattern set. First, it was shown that the maximum recall on the entity extraction task by taking all NE instances is 83%, the rest of the instances cannot be captured because of NE recognition problems. While the current subtree-model achieves 54%, potentially, using only WordNet, the expanded pattern set achieved 68% recall. The combination of both replacement methods achieved 69%.

However, in practice, the size of the expanded pattern set is 10 times bigger than the original pattern set, which is likely to decrease precision. It is necessary to find an efficient way to identify the relevant expanded patterns, and we leave this as future work.

On-Demand Information Extraction

For addressing a broader range of IE tasks, the customization of IE system emerges as a central topic of research. On-Demand Information Extraction (ODIE) is a further enhancement of such customization. In the ODIE framework, the system takes the query from the user and extracts the relevant events into the template fully automatically.

Surdeanu and Harabagiu first discussed such a system in [38]. Their "Open-*Domain* Information Extraction" system assumed the relevant document set as a system input, while in the ODIE framework the system uses an information retrieval module to obtain a relevant document set. Thus, for the ODIE framework, the document set is assumed to be noisier, and, therefore, a more robust extraction technique is needed.

The QDIE system fits in the ODIE framework as a pattern discovery component. The QDIE component provides the set of extraction patterns for the user's query. The slot assignment which is left to the user in the pre-CODIE system (see Chapter 5) can be automated by identifying sets of extraction patterns that are semantically equivalent. Paraphrase discovery is an emerging field (e.g. [6, 32]) and especially, Shinyama and Sekine [32] show phrase-level paraphrase for information extraction. Such technology is an ideal candidate for grouping extraction patterns.

Figure 6.2 shows a general ODIE architecture. The event detector module merges the result of pattern matches from different pattern groups into events, which will be the final output of the ODIE system.

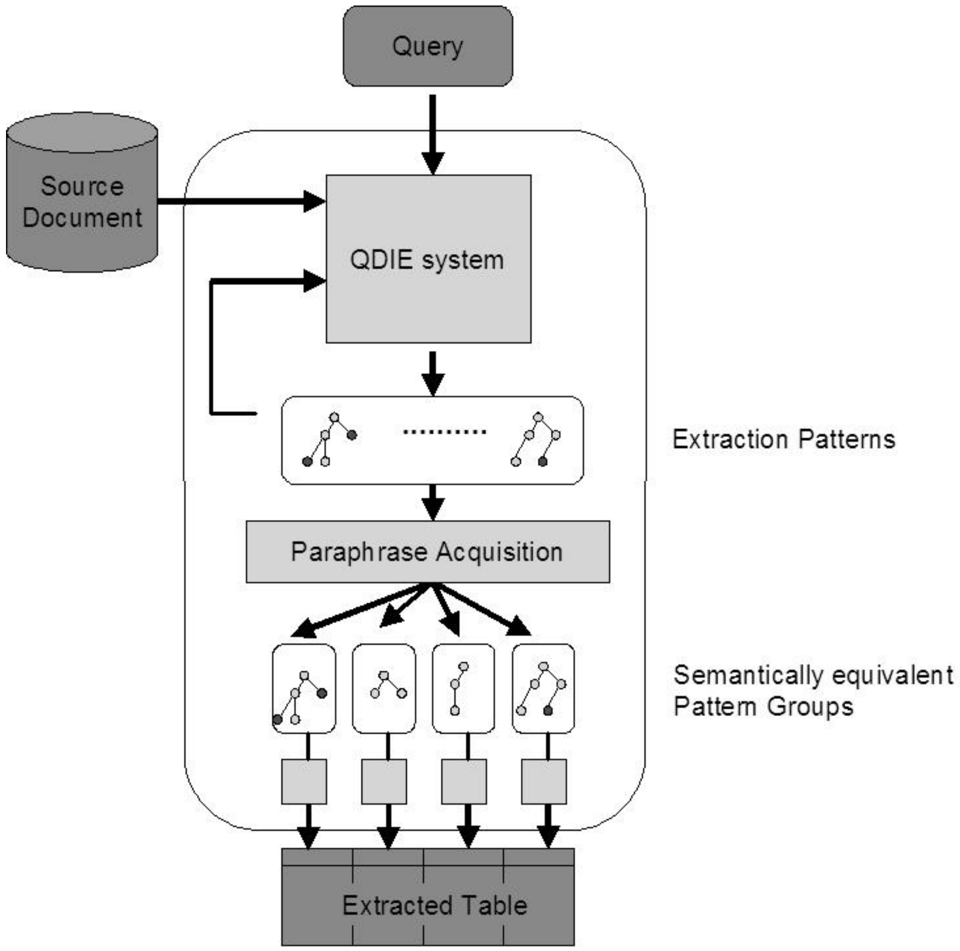


Figure 6.2: Data-flow Diagram for On-Demand Information Extraction system:

Bibliography

[1] *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Columbia, MD, November 1995. Morgan Kaufmann.

[2] *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, Fairfax, VA, 1998. http://www.itl.nist.gov/iaui/894.02/related_projects/muc/.

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

[4] Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plaintext collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries*, 2000.

[5] D. Ayuso, S. Boisen, H. Fox, H. Gish, R. Ingria, and R. Weischedel. BBN PLUM: Description of the PLUM System as Used for MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 169–176, 1992.

[6] Regina Barzilay and Kathleen McKeown. Extracting paraphrases from a parallel corpus. In *Proceedings of the 38th Meeting of the Association for Computational Linguistics*, Toulouse, France, 2001.

[7] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 1998 Conference on Computational Learning Theory*, 1998.

[8] Sergey Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, EDBT'98*, 1998.

[9] Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

[10] Fouad Soufiane Douzidia and Guy Lapalme. Lakhas, an Arabic summarization system. In *Proceedings of DUC2004*, 2004.

[11] Jerry R. Hobbs, Douglas Appelt, Mabry Tyson, John Bear, and David Islael. SRI International: Description of the FASTUS System Used for MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 268–275, 1992.

[12] Scott. B. Huffman. Learning information extraction patterns from examples. *Lecture Notes in Computer Science*, 1040:246–??, 1996.

[13] IBM. *Internet Honyaku-no Osama Bilingual (Internet King of Translation Bilingual).* http://www-6.ibm.com/jp/software/internet/king/.

[14] Jun-Tae Kim and Dan I. Moldovan. Lexical knowledge acquisition for knowledge-based information extraction. *IEEE Transactions on Knowledge and Data Engineering*, pages 713–724, October 1995.

[15] G. Krupka, P. Jacobs, L. Rau, L. Childs, and I. Sider. GE NLTOOLSET: Description of the System as Used for MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 177–185, 1992.

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

[17] Sadao Kurohashi and Makoto Nagao. Kn parser : Japanese dependency/case structure analyzer. In *Proceedings of the Workshop on Sharable Natural Language Resources*, 1994. http://www-nagao.kuee.kyoto-u.ac.jp/nl-resource/knp-e.html.

[18] Leah Larkey and Margaret Connell. Structured Queries, Language Modeling, and Relevance Modeling in Cross-Language Information Retrieval. In *Information Processing and Management, Special Issue on Cross Language Information Retrieval*, 2003.

[19] W. Lehnert, C. Cardie, D. Figher, J. McCarthy, E. Riloff, and S. Soderland. University of massachusetts: Description of the circus system as used for muc-4. In *Proceedings of the Fourth Message Understanding Conference (*MUC-4*)*, pages 282–288, 1992.

[20] A. Meyers, R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman. Annotating Noun Argument Structure for NomBank. In *Proceedings of LREC-2004*, Lisbon, Portugal, 2004.

[21] A. Meyers, R. Reeves, and Catherine Macleod. NP-External Arguments: A Study of Argument Sharing in English. In *The ACL 2004 Workshop on Multiword Expressions: Integrating Processing*, Barcelona, Spain, 2004.

[22] R. Michalski. A theory and methodology of inductive learning. *Artificial Intelligence*, 20, 1983.

[23] Tom Mitchell. Generalization as search. *Artificial Intelligence*, 18, 1982.

[24] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[25] Ellen Riloff. Automatically constructing a dictionary for information extraction tasks. In *National Conference on Artificial Intelligence*, pages 811–816, 1993.

[26] Ellen Riloff. Automatically generating extraction patterns from untagged text. In *AAAI/IAAI, Vol. 2*, pages 1044–1049, 1996.

[27] Ellen Riloff. An empirical study of automated dictionary construction for information extraction in three domains. *Artificial Intelligence*, 85(1–2):101–134, 1996.

[28] Ellen Riloff and Rosie Jones. Learning Dictionaries for Information Extraction by Multi-level Boot-strapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 1044–1049. The AAAI Press/MIT Press, 1999.

[29] Ellen Riloff, Charles Schafer, and David Yarowsky. Inducing information extraction systems for new languages via cross-language projection. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, 2002.

[30] Satoshi Sekine and Chikashi Nobata. Definition, Dictionary and Tagger for Extended Named Entities. In *Proceedings of the Forth International Conference on Language Resources and Evaluation (LREC-2004)*, 2004.

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

[32] Yusuke Shinyama and Satoshi Sekine. Paraphrase acquisition for information extraction, 2003.

[33] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1):233–272, 1999.

[34] Stephen Soderland, David Fisher, Jonathan Aseltine, and Wendy Lehnert. CRYSTAL: Inducing a conceptual dictionary. In Chris Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1314–1319, San Francisco, 1995. Morgan Kaufmann.

[35] Kiyoshi Sudo. Japanese information extraction with automatically extracted patterns. In *Proceedings of Student Research Workshop at the 39th Annual Meeting of the Association for Computational Linguistics*, Toulouse, France, 2001.

[36] Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. Automatic pattern acquisition for japanese information extraction. In *Proceedings of Human Language Technology Conference (*HLT2001*)*, San Diego, CA, 2001.

[37] Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. Cross-lingual Information Extraction System Evaluation. In *Proceedings of the 20th International Conference on Computational Linguistices (COLING-04)*, 2004.

[38] Mihai Surdeanu and Sanda Harabagiu. Infrastructure for Open-Domain Information Extraction. In *Proceedings of the Human Language Technology Conference (HLT-2002)*, 2002.

[39] Roman Yangarber. *PET: The Proteus Extracton Toolkit. User and Developer Manual*, 1997.

[40] Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. Automatic acquisition of domain knowledge for information extraction. In *In Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, Saarbrucken, Germany, August 2000.

[41] Ying Zhao and George Karypis. Criterion Functions for Document Clustering: Experiments and Analysis. Technical Report 01-40, Department of Computer Science, University of Minnesota, 2001.

[42] Ying Zhao and George Karypis. Hierarchical Clustering Algorithms for Document Datasets. Technical Report 03-027, Department of Computer Science, University of Minnesota, 2003.