Record Extraction from Large-scale Text Resources Considering Topics

Jianwei Zhang†

In recent years, the research of record extraction from large document data is becoming popular. However there still exist some problems in record extraction. 1) when large document data is used for the target of information extraction, the process usually becomes very expensive. 2) it is also likely that extracted records may not pertain to the user's interest on the aspect of the topic. To address these problems, in this paper we propose a method to efficiently extract those records whose topics agree with the user's interest. To improve the efficiency of the information extraction system, our method identifies documents from which useful records are probably extracted. We make use of user feedback on extraction results to find topic-related documents and records. Our experiments show that our system achieves high extraction accuracy across different extraction targets. In our demo, we will demonstrate how record extraction process and document selection process are going, and show extraction results from documents respectively identified by four different methods.

1. Introduction

With the recent progress of information delivery services, electronic text data is increasing rapidly. Useful information often exists in these text documents. However, computers can not easily process the information because it is usually hidden in unstructured texts. Therefore *information extraction* is becoming an important technique to find useful information from a large amount of text documents. Especially a lot of researches analyze the document structures and contexts to construct relational tables. Among many approaches, the *bootstrapping* extraction methods[1),2)] have attracted a lot of research interests. These approaches expand the target relation by exploiting the duality between patterns and relations starting from only a small sample. The extracted information, which we call *records*, can be used as a relational table for answering SQL queries or being integrated with other databases.

Two problems exist in the previous approaches of information extraction. In general, an information extraction system needs to preprocess the documents (e.g. attaching named entity tagger to recognize person names, organization names and location descriptions etc.) and scan the documents. First, when the text document set is very large, processing all the documents is quite expensive. Second, records whose topics are not desirable for a user may also be extracted by only using pattern matching. For example, for a user who wants to acquire the information of IT companies and their locations, he is not satisfied with other topic-unrelated pairs (e.g. automobile companies and locations).

To solve these two problems, we propose a method to efficiently extract information suitable for the user's intention. In general, only a part of the documents in a large data set is useful for the extraction task. We manage to specify the documents that are likely to contain desirable records as the target documents for extraction. The efficiency is improved by processing not all the documents, but just a subset of them. From the selected documents related to the required topic, topic-related records are more likely to be extracted.

The rest of this paper is structured as follows. Section 2 reviews the related work. In section 3, the overview of the proposed system is first presented, and then the procedure for extracting records and several document selection methods are described. Section 4 shows the experimental results and their evaluations. Our demonstration scenario is described in Section 5. Finally, we conclude this paper and discuss the future work in Section 6.

2. Related Work

There have been many researches on information extraction from unstructured and semi-structured documents such as Web and news

† Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba

archives. Craven et al.[3] develop an information extraction system to create a knowledge base from the Web by using machine learning to exploit classes and relations, given an ontology and training data in advance. *Lixto*[4] is a visual web information extraction system that allows a user to specify the extraction patterns. Kushmerick[5] applies a machine learning method to learn extraction rules, given a set of manually labeled pages. As opposed to these approaches dealing with one web page or some similarly structured web pages, *bootstrapping methods*[1),2),6)] are proposed to extract information from documents whose structures are very different in a scalable manner. *DIPRE* (Dual Iterative Pattern Relation Extraction)[1] exploits the duality between patterns and relations from web pages. For example, beginning with a small seed set consisting of several (author, title) pairs, DIPRE generates patterns which are used to find new books. This technique is proved that it works well because relation pairs tend to appear in similar contexts in the Web environment. Zhang et al.[6] propose a method to estimate the coverage of extracted records to reduce iterations of extraction loops, and a technique to estimate the error rate of extracted information to improve the extraction quality. *Snowball*[2] considers the problem of extracting relation pairs from plaintext documents. This method improves the DIPRE method by using novel pattern representation including named entity tags, and precise evaluation measure of patterns and records so that more reliable results can be extracted. In this paper, we extend the basic framework of DIPRE and Snowball methods for the record extraction.

QProber[7] uses a small number of query probes to automatically classify hidden web databases. Chakrabarti et al. propose a topic-focused web crawling method through relevance feedback[8]. The focused crawler based on a hypertext classifier[9] classifies crawled pages with categories in a topic taxonomy. We take the hint from these researches to prefer selecting useful documents as extraction targets, just as the focused crawler fetches relevant web pages and discards irrelevant ones. Agichtein et al. present a method[10] to retrieve documents and from them extract information from an efficiency viewpoint, but they do not consider whether the extraction results satisfy the user's interest with respect to the topic. In the best of our knowledge, our system is the first to provide topic-related information extraction facility using an interactive approach.

3. Record Extraction Incorporating Document Selection

3.1 System Overview

In this section, we describe the proposed system architecture (Figure 1).

In this paper, we also focus on the problem of extracting a relation of companies and their locations defined in Snowball[2]. Different from its scenario, we consider a user usually prefers the extraction results on a specific topic (e.g., "IT" companies and their locations), to all the extractable records. Extracted pairs on other topics are unwantedly troublesome. We present a method to solve this problem. Suppose that several samples (Seed Record Set in Figure 1) are given as initial knowledge and they also reflect the topic that a user is interested in.

Document repository is large collections of text documents such as newspaper archives. *Document Set for Record Extraction* (DSRE) is a subset of document repository, consisting of documents where records on the related topic may exist. We index the documents using a full-text search system so that given appropriate queries the corresponding documents can be returned. In our proposed system, not all the documents in the document repository are scanned at one time by the extraction system. Considering that records related to the specific topic are more likely to be extracted from the topic-related documents, those documents are preceded as the extraction target. The efficiency is improved by only processing the documents worthy of analysis. Initially the DSRE set can be retrieved by using the sample records as the query. This DSRE set is continuously extended with the newly selected documents, i.e. the output of *Document Selection Module* to be explained later.

Record Extraction Module is to extract records from the DSRE set. We extend the bootstrapping framework of the DIPRE and Snowball systems. Beginning with the seed records given from a user, the program finds the

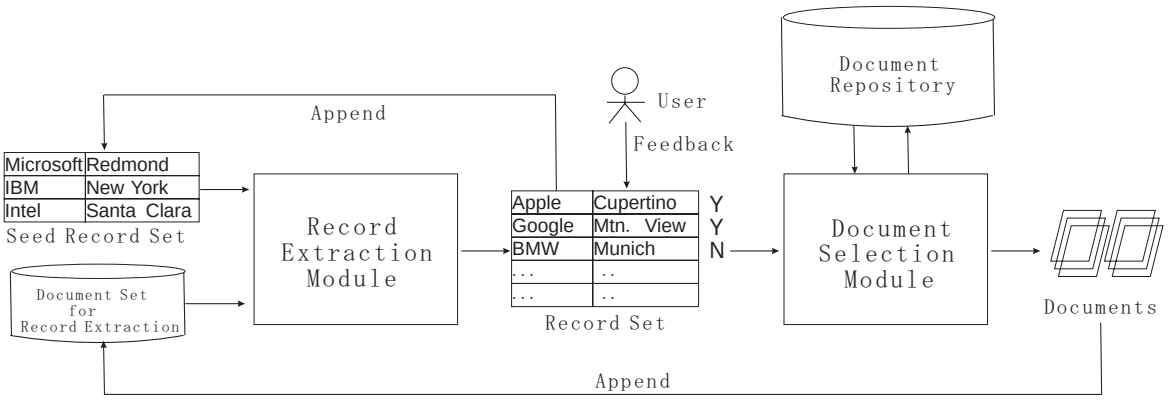

Fig. 1 System Components

occurrences of those records. Then the occurrences are analyzed to generate patterns. Using the patterns, the program searches the documents to match new records. This process is repeated until some termination criterion is met. In this way, a number of records can be obtained with a minimal sample from the user. We describe the details in section 3.2.

Next we consider the *User Feedback* process occurs after the records are extracted. Because the number of extracted records is generally large, it is not feasible that the user judges all of them. Therefore it is necessary to lighten the user's work. For the user, records with no or little noise are apt to judge. Thus we sort the extracted records in terms of a reliability measure so that the reliable ones are brought to the top place. What the user has to do is to check only the top ranked ones.

There are five kinds of feedback on the extracted records.

(1) Desirable Record: The records have no noise, correct corresponding relationship and related topic.

(2) Unrelated Topic: Although the extracted records are the right pairs of valid companies and locations, their topics are not what the user is concerned about. For example, for the user who is interested in the IT information, the (*BMW*, *Munich*) pair is not satisfactory and marked as "Unrelated Topic".

(3) Incorrect Tag Recognition: Company names and location descriptions in the extracted records may not be valid due to wrong entity assignment from a named entity tagger. In the experiments described later, we observed some noisy pairs such as (*Com Corp.*, *Santa Clara*), (*Cupertino*, *Calif.*) were also extracted. The named entity tagger should take the blame for the misidentified companies and locations. In the above examples, 3*Com Corp.* was intercepted as *Com Corp.* and *Cupertino* that is really a city name was mistaken for a company name.

(4) Wrong Relation: This means although both company and location are valid ones, the location is the place where the company is located in.

(5) Unknown: The user can not judge whether the extracted record is a desired one.

Based on the experiments, we observed that the third and fourth cases rarely appeared in the top ranked records after we sorted them. Therefore, we do not discuss those two cases too much, and pay more attention to the first and second kinds of feedback to select documents for extraction.

Document Selection Module receives the user feedback and selects documents useful for extraction from the document repository. The selected documents are appended to the DSRE set for subsequent record extraction. In section 3.3, we describe four methods of document selection in detail.

3.2 Record Extraction

For record extraction, our approach is based on the bootstrapping approach (Algorithm 1). In this algorithm, the document repository may be considered as static and relatively small. We extend the process of record extraction incorpo-

rating the document selection process in next section.

Algorithm 1 Record Extraction Based on Bootstrapping Approach

1: $Seed$: *Seed Record Set*
2: Doc : *Document Repository*
3: $Doc\_tag = attach\_tag(Doc)$
4: **repeat**
5: $Occ = find\_occurrences(Doc\_tag, Seed)$
6: $Pat = generate\_patterns(Occ)$
7: $Rec = extract\_records(Doc\_tag, Pat)$
8: $Top\_Rec = sort\_records(Rec)$
9: $Seed = Seed + Top\_Rec$
10: **until** *termination criterion is met*
11: **return** *Rec*

The process flow is as follows.

(1) **Providing Seed Records and Attaching Named Entity Tags**: A seed record set (e.g., the example in Figure 1) is first given by a user. This set should declare the target relation the user wants to obtain and reflect the topic he cares for (e.g., he is interested in IT companies and their locations). As a preprocessing, we can use a named entity tagger to recognize person, organization, location, etc. occurring in the documents.

(2) **Finding Occurrences**: Then we find *occurrences* of the records in the seed record set from the document repository. Occurrences are the contexts surrounding the attributes of the records. They are defined as the following style for our example case:

$(comp, loca, o\_prefix, tag1, o\_middle, tag2, o\_suffix)$

, where *comp* is a company name and *loca* represents its location. For the seed set in Figure 1, *comp* and *loca* correspond to $Microsoft$ and *Redmond*, respectively. o_prefix is the context preceding the attribute (*comp* or *loca*) appearing first, and o_suffix is the context following the last attribute. o_middle is the string between two attributes. $tag1$ and $tag2$ are the named entity tags. For the task in this paper, we pay attention to the ORGANIZATION and LOCATION tags. Other kinds of tags, such as PERSON, are not considered.

(3) **Generating Patterns**: Next patterns are generated by analyzing the occurrences. Patterns are defined as the following style:

$(p\_prefix, tag1, p\_middle, tag2, p\_suffix)$

. First the occurrences are partitioned into groups. The occurrences in each group have the same $tag1$, o_middle and $tag2$. If the number of occurrences in a group is less than a threshold, the group is deleted. Patterns are generated for the remaining groups. The $tag1$, p_middle and $tag2$ of a pattern is same with those of the occurrences in the group. For each group, the longest common suffix of all the $o_prefix$es becomes the p_prefix of the pattern, and the longest common prefix of all the $o_suffix$es is the p_suffix of the pattern.

(4) **Extracting Records**: Using the generated patterns in the previous step, the document set is scanned again to find new records matching the patterns.

(5) **Sorting Records**: For selecting new records to be appended to the seed set and picking up records to receive feedback from the user, we sort the extracted records. Generally the probability that a record has noise is small if it is extracted by multiple patterns. Furthermore the more documents an extracted record appears in, the more reliable it is. Thus we give the order to the extracted records according to their numbers of patterns and numbers of documents where they occur. That is to say, records are first sorted in the descending order of the numbers of patterns extracting them, and in the case that the numbers of patterns are equivalent, the numbers of documents containing them are then considered.

The top ranked records are used as the new seeds and then a new loop is begun. This repeated procedure terminates until a given condition is met (e.g., convergence, which means no more records can be extracted). In this way, a large amount of records can be obtained starting from a small sample set.

3.3 Document Selection

The previous extraction technique is supposed to examine all the documents in the document repository. When the repository is very large, the extraction process is time consuming. It is also unavoidable to extract undesired records from unrelated documents. Therefore we consider choosing documents for desirable extraction. The process of record extraction combined with document selection is shown in Algorithm 2. The main difference from Algorithm 1 described in Section 3.2 is that the document selection (Step 5-7) is done before the record extraction. At each iteration of the repeated procedure, new documents are chosen and only these documents are passed through a named entity tagger. The same documents are processed by the named entity tagger only once. Receiving feedback from a user (Step 12) is performed only for the third and fourth methods described later, not for the first and second one. The termination criterion may be that a certain number of selected documents are reached, and based on them the extraction process converges.

Algorithm 2 Record Extraction Incorporating Document Selection

1: $Seed$: *Seed Record Set*
2: Doc : *Document Repository*
3: $Doc\_tag = \phi$: *Tagged Documents Set*
4: **repeat**
5: $D = select\_documents(Doc)$
6: $D\_tag = attach\_tag(D)$
7: $Doc\_tag = Doc\_tag + D\_tag$
8: $Occ = find\_occurrences(Doc\_tag, Seed)$
9: $Pat = generate\_patterns(Occ)$
10: $Rec = extract\_records(Doc\_tag, Pat)$
11: $Top\_Rec = sort\_records(Rec)$
12: $Top\_Rec = review\_feedback(Top\_Rec)$
{This step may be disregarded for different document selection methods}
13: $Seed = Seed + Top\_Rec$
14: **until** *termination criterion is met*
15: **return** Rec

We assume that the document repository is indexed. Given a query, corresponding documents can be retrieved. For comparison, we present four methods of document selection. In the rest of this section, we discuss how they work.

(1) **Baseline**: This simplest method is to choose documents randomly as the target of extraction from the document repository.

(2) **Records without Feedback**: This method simply employs the words appearing in the top ranked records as the query. The query is composed of the disjunction of the attribute values of the records. For the extraction example in Figure 2, the query is "(*Apple* AND *Cupertino*) OR (*Google* AND *Mtn.* AND *View*) OR (*BMW* AND *Munich*) OR (*NEC* AND *Tokyo*)".

Company	Location
Apple	Cupertino
Google	Mtn. View
BMW	Munich
NEC	Tokyo
⋮	⋮

Fig. 2 Extraction Example

Company	Location	Feedback
Apple	Cupertino	Yes
Google	Mtn. View	Yes
BMW	Munich	No
NEC	Tokyo	Yes
⋮	⋮	⋮

Fig. 3 User Feedback

(3) **Records with Feedback**: Our experimental observation is that most of the top ranked records used as the query in the previous method are noiseless ones. However the records on different topics sometimes come to the top place. For example, the (BMW, Munich) pair is unsatisfactory for a user who wants to obtain the IT information. If the records on inappropriate topic are popular, many topic-unrelated documents may be retrieved and in turn undesired records are extracted from these documents. In the third method, we consider eliminating the records on different topics with the help of the user. A user gives his feedback about whether a record is topic-related or not. Only the records judged as good ones by

the user are used as the query. For the exaction example in Figure 3, the query is "(*Apple* AND *Cupertino*) OR (*Google* AND *Mtn. View*) OR (*NEC* AND *Tokyo*)", where (*BMW*, *Munich*) is not contained.

(4) **Learning**: In the fourth method, the top ranked records also receive the feedback from the user. At the next step unlike the above methods in which the words of records are directly used as the query, we manage to identify feature words that represent the concerned topic as the query. Based on the feedback results, we first select a training document set consisting of relevant documents and irrelevant ones. Then the training document set is used to generate an ordered list of words appearing in the relevant documents. The top ranked words tend to represent the topic that a user is interested in. The disjunction of top k words constitutes the query.

(a) Selecting relevant and irrelevant documents

First the documents from which more than one record are extracted are detected as *Relevant Document Candidates* (RDC). Then we assign scores to the documents in the RDC set using the following formula:

$$score(d) = \frac{r+pu}{r+w+u} * \log(r + pu + 1)$$

where d represents a document in RDC, r is the number of records on right topic, w is the number of records on wrong topic, u is the number of records whose topics a user did not (or could not) decide, and p is the probability that an unjudged record agrees with the concerned topic. For different tasks, p may be given a different value empirically. The top n documents with the highest scores are used as the *relevant documents*. In this way, the relevant documents tend to be the ones from which many desirable records are extracted and the ratios of them to all the extracted records are high. We also randomly select the documents that do not overlap the RDC set as the *irrelevant documents*.

(b) Learning feature words

Then each word t in the relevant documents is assigned the Okapi[11] weight:

$$score(t) = \frac{(r_t+0.5)/(R-r_t+0.5)}{(n_t-r_t+0.5)/(N-n_t-R+r_t+0.5)}$$

where r_t is the number of relevant documents containing t, n_t is the number of documents containing t, R is the number of relevant documents, and N is the number of documents in both the relevant document set and the irrelevant document set. Intuitively the word t that appears in many relevant documents and rarely appears in the irrelevant documents, can get a high score. We observed that most of the top ranked words were the names of popular companies and their locations, or the words representing the concerned topic in our experiments.

4. Experiments

4.1 Experimental Setting

For our experiments, we use the document repository of Wall Street Journal from 1986 to 1992 consisting of 173,039 documents. For named entity recognition, we use the named entity tagger released by University of Illinois[12]. It can recognize PERSON, LOCATION, ORGANIZATION and MISC. entities in English. For the information retrieval system[13], we use a full-text search engine *Namazu*[14], that is popular in Japan. Namazu supports a Boolean retrieval model with tf-idf ranking, which simply adds up the tf-idf values of the words appearing in the query for each document as the score of the document.

4.2 Experimental Results

In this section, we introduce two extraction targets on different topic. We experimentally compare the extraction results performed on the documents selected by the four different document selection methods.

4.2.1 Extraction of IT companies and locations

For this target, we assume a user provides five

Table 1 Example Records for IT Target

IT Company	Location
Xerox	Stamford
Intel	Santa Clara
Apple	Cupertino
Compaq	Houston
Sun	Mountain View

Table 2 The Number of Patterns and Records for IT Target

	Baseline	Rec without Fdbk	Rec with Fdbk	Learning
#patterns	2	9	13	10
#records	130	2800	3815	2909

Table 3 Extraction Quality for IT Target

	Baseline	Rec without Fdbk	Rec with Fdbk	Learning
#records without noise	48	47	50 (43)	50 (49)
#records on IT topic	9	20	44 (33)	41 (37)

examples as Table 1 shows. We select 5000 documents from the document repository respectively using the four document selection methods. The extraction results are shown in Table 2. From the documents selected by the *baseline* method, few patterns are generated and consequently not so many records are extracted. This is because the probability that randomly selected documents may contain occurrences and patterns of records is relatively small. In contrast, about 10 patterns are generated and about 3000 records are extracted from other three 5000 documents set chosen by other document selection methods.

We sort the extracted records and manually evaluate the top ranked 50 ones. Table 3 shows the evaluation results. The numbers in the first row are those of records without noise among the checked 50 records. The second row represents the numbers of topic-related records among the records without noise. As we can see, less than one half of records has the concerned topic for the *Baseline* and *Records without Feedback* methods, while the ratios of the records on the desirable topic extracted from the documents selected by the *Record with Feedback* and *Learning* methods are much higher. Notice that the document selection methods of *Records with Feedback* and *Learning* require the user's feedback. The top ranked records in the extraction results may also include the ones that have received feedback from the user. After eliminating the records that have been judged midway, the evaluation results are reported in the brackets. For example, after sorting the records extracted from the 5000 documents selected by the *Records with Feedback* method, we pick up the top 50 records that the user did not give his feedback and evaluate them. Among the 50 records, 43 ones have no noise and the 43 records include 33 IT pairs.

In summary, *Records without Feedback*, *Records with Feedback* and *Learning* tend to find the documents where more patterns and records can be generated than *Baseline*. With feedback incorporated, *Records with Feedback* and *Learning* help select useful documents and feed them to the extraction system so that more topic-related records are recognized than *Baseline* and *Records without Feedback*.

4.2.2 Extraction of Biotechnology companies and locations

We also do experiments for another topic of biotechnology to further test the generality of different document selection methods' effects. Because the biotechnology topic is not as popular as the IT topic, we limit the number of documents as the extraction target to 1000. For this target, the seed record set (Table 4) also consists of only five pairs of biotechnology companies and locations. Extraction results and qualities are shown in Table 5 and Table 6. As we can see, they have the trends similar to the case of the IT topic.

4.2.3 Discussion

As we can see, the numbers of patterns and records (Table 2 and Table 5) extracted from the documents chosen by the *Records with Feed-*

Table 4 Example Records for Bio Target

Bio Company	Location
Amgen	Thousand Oaks
Genentech	South San Francisco
Biogen	Cambridge
Chiron	Emeryville
Gilead Sciences	Foster City

Table 5 The Number of Patterns and Records for Bio Target

	Baseline	Rec without Fdbk	Rec with Fdbk	Learning
#patterns	2	8	10	10
#records	6	1103	743	714

Table 6 Extraction Quality for Bio Target

	Baseline	Rec without Fdbk	Rec with Fdbk	Learning
#records without noise	6	29	29 (23)	30 (30)
#records on bio topic	2	15	28 (10)	24 (17)

back and *Learning* methods are close, and their qualities (Table 3 and Table 6) nearly draw. However, for obtaining the same number of documents, the user's labors that two methods require are quite different. In the IT experiments, for the document selection method of *Records with Feedback*, the user looks through the sorted records from the top ones at each iteration, and totally gives 124 feedback, which includes 75 desirable records and 49 ones with noise or on unrelated topic. The *Records with Feedback* method uses the records as the query. Therefore for obtaining a relatively large number of documents, enough judged records are indispensable. In contrast, the *Learning* method uses the feature words representing the concerned topic as the query so that selecting a specific number of documents does not directly depend on the feedback number (i.e., the number of records judged as desirable). For the *Learning* method, the user feedback is only used to choose the relevant documents to construct the training data. The learning result, a ranked word list, is used to generate the query for retrieving documents. The requirement for a larger number of documents can be solved by expanding the query with the disjunction of more feature words, not by checking more records. Actually the feedback number for the *Learning* method is 21, which is even smaller than 124 of *Records with Feedback*, but causes the rivalrous extraction results.

5. Demonstration Scenario

In this demonstration, we show a prototype of the REIDS (Record Extraction Incorporating Document Selection) system, including the following highlights:

(1) Four Document Selection Options: We implement four optional methods of document selection described in Section 3.3. A user can select one of four methods and specify the number of documents to be used as extraction target. Figure 4 is the prototype snapshot.

(2) Extraction Information Exhibition: Extraction results from documents identified by different methods can be displayed. Extraction information summaries, including the numbers of retrieved documents, generated patterns and extracted records, are shown. A user can further browse the appearances of real patterns and records by clicking the corresponding links. Two snapshots of extraction results are shown in Figure 5 and Figure 6.

(3) User Feedback Interface: For the methods which require user feedbacks, we provide an interface to receive inputs from a user. The user can give the system his judgement about whether an extracted record fits his interest. Then the system receives the user feedbacks and consequently selects suitable documents to feed them to record extraction process. Figure 7 snapshoots the user feedback interface.

Fig. 4 System Prototype Snapshot

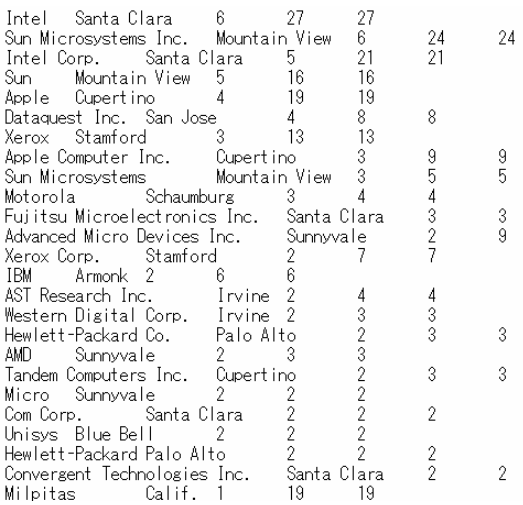

Fig. 6 Extracted Records Snapshot

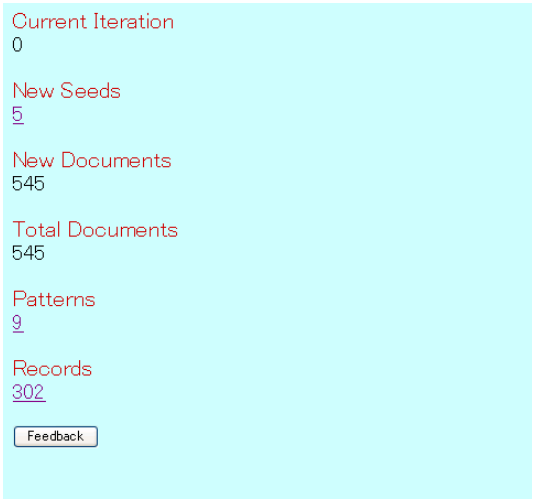

Fig. 5 Extraction Information Summary Snapshot

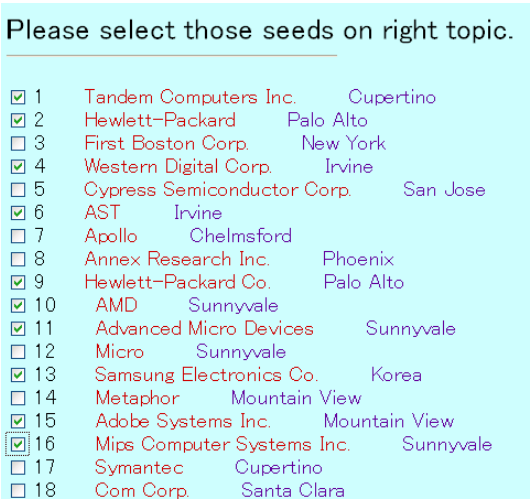

Fig. 7 User Feedback Interface Snapshot

6. Conclusions and Future Work

In this paper, we proposed a record extraction method incorporated with document selection to efficiently acquire topic-related records. We showed the significant improvement of extraction qualities by using feedback to select documents as extraction target.

The current experiments are restricted in the extraction of (company, location) pairs. In our future work, we will make more attempts to extract other relations. It is also an interesting work to put the effect of feedback on pattern and record evaluation. Moreover not only the feedback from a user but also the integration of extraction results with other existent databases is also considerable. Our ongoing research will address these questions.

Acknowledgments This work was supported by a grant-in-aid for Initiatives for Attractive Education in Graduate Schools "the educational program of advanced informatics human resources development by the provision with practical IT ability" from MEXT. I thank Hiroyuki Kitagawa and Yoshiharu Ishikawa for their helpful guidance and constructive comments. I also thank Sayumi Kurokawa for her fruitful discussions.

References

1) S. Brin, Extracting Patterns and Relations from the World Wide Web. *Proc. WebDB*, 1998.

2) E. Agichtein and L. Gravano, Snowball: Extracting Relations from Large Plain-Text Collections. *Proc. ACM SIGMOD*, 2001.

3) M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam and S. Slattery, Learning to Construct Knowledge Bases from the World Wide Web. *Artificial Intelligence*, Vol. 118, No. 1-2, pp. 69-113, 2000.

4) R. Baumgartner, S. Flesca and G. Gottlob, Visual Web Information Extraction with Lixto. *Proc. VLDB*, pp. 119–128, 2001.

5) N. Kushmerick, Wrapper Induction: Efficiency And Expressiveness. *Artificial Intelligence*, Vol. 118, No. 1-2, pp. 15–68, 2000.

6) R. Y. Zhang, L. V.S. Lakshmanan and R. H. Zamar, Extracting Relational Data from HTML Repositories. *SIGKDD Explorations*, 2004.

7) L. Gravano, P. Ipeirotis and M. Sahami, QProber: A System for Automatic Classification of Hidden-web Databases. *ACM Trans. Inf. Syst.*, Vol. 21, No. 1, pp. 1–41, 2003.

8) S. Chakrabarti, K. Punera and M. Subramanyam, Accelerated Focused Crawling through Online Relevance Feedback. *Proc. WWW*, pp. 148–159, 2002.

9) S. Chakrabarti, M. van den Berg, and B. Dom, Focused Crawling: A New Approach to Topic-specific Web Resource Discovery. *Computer Networks*, Vol. 31, No. 11-16, pp. 1623-1640, 1999.

10) E. Agichtein and L. Gravano, Querying Text Databases for Efficient Information Extraction. *Proc. ICDE*, pp. 113–124, 2003.

11) S. E. Robertson, Overview of the Okapi projects. *Journal of the American Society for Information Science*, Vol. 53, No. 1, pp. 3–7, 1997.

12) Named Entity Tagger: http://l2r.cs.uiuc.edu/cogcomp/asoftware.php?skey=NE

13) G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, 1983.

14) Namazu: http://www.namazu.org/index.html.en