A Practical QA System in Restricted Domains

Hoojung Chung, Young-In Song, Kyoung-Soo Han, Do-Sang Yoon, Joo-Young Lee, Hae-Chang Rim
Dept. of Comp. Science and Engineering
Korea University
Seoul 136-701 Korea
{hjchung,sprabbit,kshan,yds5004,jylee,rim}@nlp.korea.ac.kr

Soo-Hong Kim
Dept. of Comp. Software Engineering
Sangmyung University
Chonan 330-720 Korea
soohkim@smuc.ac.kr

Abstract

This paper describes an on-going research for a practical question answering system for a home agent robot. Because the main concern of the QA system for the home robot is the precision, rather than coverage (No answer is better than wrong answers), our approach is try to achieve high accuracy in QA. We restrict the question domains and extract answers from the pre-selected, semi-structured documents on the Internet. A named entity tagger and a dependency parser are used to analyze the question accurately. User profiling and inference rules are used to infer hidden information that is required for finding a precise answer. Testing with a small set of queries on weather domain, the QA system showed 90.9% of precision and 75.0% of recall.

1 Introduction

During the last decade, automatic question-answering has become an interesting research field and resulted in a significant improvement in its performance, which has been largely driven by the TREC (Text REtrieval Conference) QA Track (Voorhees, 2004). The best of the systems in the QA Track is able to answer questions correctly 70% of the time (Light et al., 2003). The 70% of accuracy is, of course, high enough to surprise the researchers of this field, but, on the other hand, the accuracy is not enough to satisfy the normal users in the real world, who expect more precise answers.

The difficulty of constructing open-domain knowledge base is one reason for the difficulties of open-domain question answering. Since question answering requires understanding of natural language text, the QA system requires much linguistic and common knowledge for answering correctly. The simplest approach to improve the accuracy of a question answering system might be restricting the domain it covers. By restricting the question domain, the size of knowledge base to build becomes smaller.

This paper describes our restricted domain question answering system for an agent robot in home environment. One of the roles of the home agent robot is to be able to answer the practical questions such as weather information, stock quote, TV broadcasting schedule, traffic information etc. via a speech interface. The agent should provide high-precision answers, otherwise the users will not trust the entire functions of the home agent robot, which includes not only the ability of question answering but also the speech interface for controlling home appliances. That means *no answer is preferred to a wrong answer* and the primary concern in our research is improving the precision of the question answering system.

In this paper, we present a question answering system which is restricted to answer only to the questions on weather forecasts [1], and provide some experimental results of the restricted QA system. To achieve the high accuracy, the QA system processes the semi-structured text data on the Internet and store it in the form of relational database. The domain specific hand-coded ontology containing weather terms and cities is manually built for the question analysis and the inference process.

The remainder of the paper is organized as follows. Section 2 describes the overall architecture of the QA system. Section 3 describes the practical QA system. Section 4 evaluates the system and reports the limitation of the QA system. Section 5 compares our system with other QA systems. Section 6 concludes with some directions for future work.

2 Overall Architecture

The overall framework of the QA system is presented in Figure 1. The QA system consists of two major parts; the IE (Information Extractor) engine and the QA engine.

[1] We've developed the QA system for a TV broadcast schedule domain as well, which is more complex to process than the weather forecasts QA, but have not evaluated it yet. So, in this paper, we present the system for weather forecasts only.

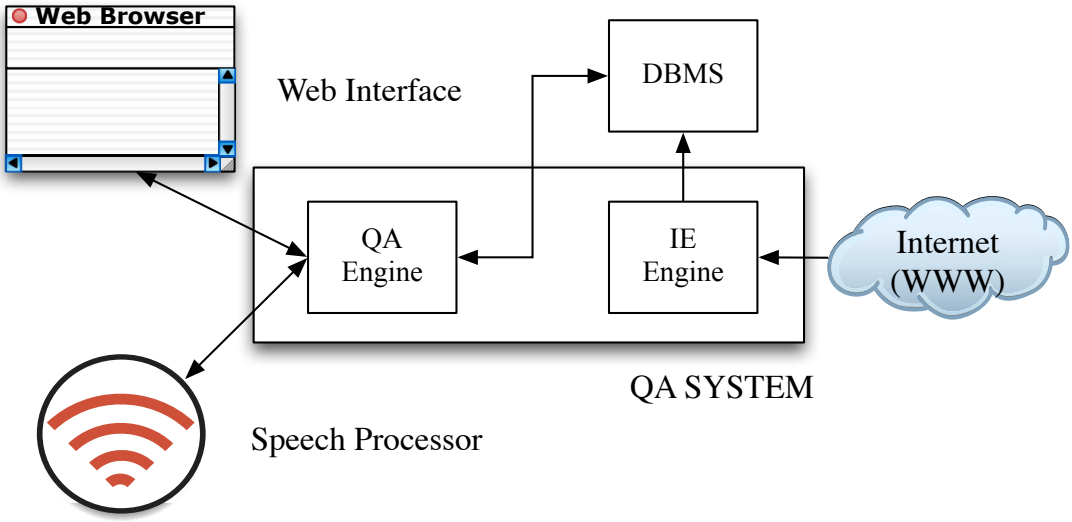

Figure 1: Overall architecture of the Question Answering System

The **IE engine** consists of two parts; a web crawler and a wrapper. The web crawler downloads the selected webpages from the website of the Korea Meteorological Administration (KMA) every hour. The website provides current weather conditions and 7 day-forecasts for dozens of Korean cities. The wrapper, which is a set of extraction rules, is used to extract weather information from the webpages . The extracted information is stored in the database.

The **QA engine** performs three-phase processing: First, it analyzes natural language questions and translates the questions into Structured Query Language (SQL) statements. Second, the SQL queries are directed to a DBMS to retrieve the answers in the database. Finally, the result from the DBMS is converted to natural language sentences for output. Figure 2 depicts overall processes for the QA engine. A DBMS (Currently, Oracle Database) is used for managing extracted data. A speech processor can be merged with the system when it is used in the home agent robot, which provides speech interface. A web interface is used for providing web-based QA service with the QA system.

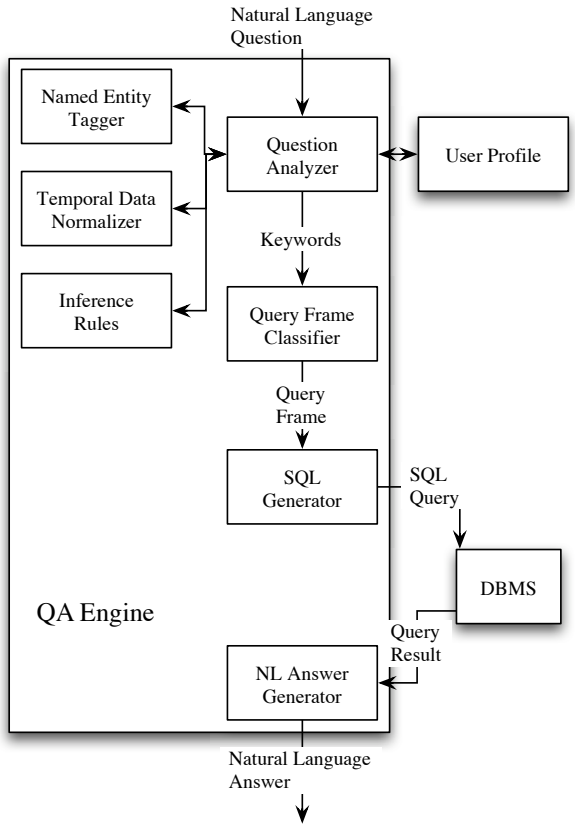

Figure 2: The QA Engine

3 A Practical QA System

The question answering starts from extracting weather information from the web site. The user request is analyzed with the question analyzer and the appropriate query frame is selected, and then the request is translated into the SQL expression. The SQL query is used to retrieve the correct answer from the database, which stores weather information from the webpages. Finally, natural language answer is generated based on the every result extracted from the DBMS.

3.1 Information Extraction

The weather information in the webpages is semi-structured. Semi-structured resources generally do not employ unrestricted natural language text, but rather exhibit a fair degree of structure (Kushmerick, 1997). Therefore, information can be accurately and easily extracted from the webpage, compared to IE from unstructured data.

On the other hand, semi-structured resources are usually formatted for use by people, and contain irrelevant elements that must be ignored, such as images, advertisements, and HTML formatting tags (Figure 3). Thus information extraction from the semi-structured documents is not entirely trivial. Currently, the QA system is using hand-coded wrappers. However, we are developing an automatic process of constructing wrappers (wrapper induction) for semi-structured resources and that can detect the modification of the web page design and adapt the wrapper according to the modification, automatically, like (Sigletos et al., 2003).

Presently, the IE engine extracts following information :

- Current observation: weather summary, visibility, temperature, wind, relative humidity
- 7 days-forecasts : weather summary, forecast temperature (highest/lowest).

3.2 Question Analysis

First, user's request is analyzed with the query analyzer as represented in Figure 2. The analyzer extracts several keywords that describing the question, such as event word, date, time, and location, by using a dependency parser, and the user question is represented only by these extracted keywords.

The named entity tagger is used to identify temporal expressions, place names, and weather events. The tagger consults the domain-dependent ontology for recognizing weather events, and the domain-independent ontology for place names. The ontology for the weather events consists of event concepts, which are similar to Synset in WORDNET (Fellbaum, 1998). For example, *rain* and *umbrella* are in same event concept in the domain ontology for weather events, because the questions about using umbrella are usually asking about raining (e.g. *Will I need to bring umbrella tomorrow?* and *Will it be raining tomorrow?*)

The temporal data normalizer converts temporal expressions such as *today*, *this weekend* and *now* into absolute values that can be used in querying to the database.

Seoul, March. 11., wide spread dust, (-/-)
Seoul, March. 12., cloudy, (0/11)
Seoul, March, 13., Sunny, (1/11)
...

Figure 3: Wrappers extracts weather information from the semi-structured documents

If the information on date, time, or location is not expressed in the user's request, the question analyzer infers the missing information. The inference rules, which are built based on our observation on various user questions, are domain-independent, because the omission of temporal or spatial information is common not only in weather information question, but also in questions for other domains.

The user profile is used for the inference in query analysis. We observed many people omit the place name in the weather-domain question. Unlike the temporal information, it is impossible to guess the current location without any user information. Thus, we store some user-related information in the user profile. Portfolio of stocks or favorite TV programs can be stored in the user profile if the QA system processes queries on stock quote or TV schedule domain.

Let's take an example of the query analysis. The following keywords are extracted from the question "Is it raining?"

```
EVENT : rain
DATE : 03/12/04
TIME : 02:20
CITY : Seoul
```

Even though the time, date, and city is not explicitly mentioned in the question, the question analyzer

infers the information with the user profile and the inference rules.

3.3 Query Frame Decision

Restricting the question domain and information resource, we could restrict the scope of user request. That is, there is a finite number of expected question topics. Each expected question topic is defined as a single *query frame*. The following are query frame examples. They are used for processing the query for the precipitation forecast for the next day, diurnal range of today, current wind direction, and current temperature, respectively.

```
[PRECIPITATION_TOMORROW]
[DIURNALRANGE_TODAY]
[WINDDIRECTION_CURRENT]
[TEMPERATURE_CURRENT]
```

Each frame has a rule for SQL generation. PRECIPITATION_TOMORROW has the following SQL generation rule.

```
[PRECIPITATION_TOMORROW]
SELECT date, amprecpr, pmprecpr FROM forecast_tbl WHERE $date $city
```

date, *amprecpr* and *pmprecpr* are field names in the database table *forecast_tbl*, which mean date, the precipitation probability of morning and afternoon of the day. The rule generates the SQL statement that means: *retrieve the precipitation probability of tomorrow morning and afternoon from the DB table which stores forecast information*.

Here is another example, which is the SQL generation rule for [DIURNALRANGE_TODAY].

```
[DIURNALRANGE_TODAY]
SELECT city, max(temp)-main(temp) FROM current_tbl WHERE $date $city group by city
```

Analyzing a question means selecting a query frame in this system. Thus, it is important to select the appropriate query frame for the user request. The selection process is a great influence on the precision of the system, while there is not much likelihood of errors in other processes, such as generating SQL query from the selected query frame, retrieving DB records, and generating an answer.

As represented in Figure 2, the extracted event, temporal and spatial keywords are used for selecting an appropriate query frame. Currently, we are using a hand-coded decision tree-like classifier for selecting an appropriate query frame for the extracted keywords. If a question isn't proper for the handling domain, the classifier rejects it. Machine learned classifier is being developed in order to substitute for the hand-coded classifier.

Is it raining?
↓
EVENT : rain
DATE : 03/12/04
TIME : 02:20
CITY : Seoul
↓
The frame [RAIN_CURRENT] is selected.
↓
SELECT time, city, weather FROM current_tbl
WHERE time='03/12/04/0200', city='Seoul'

Figure 4: Interpreting the natural language question to the SQL query

SELECT time, city, weather FROM current_tbl
WHERE time='03/12/04/0200', city='Seoul'
↓
DBMS returns "03/12/04/0200 Seoul Sunny"
↓
On 2:00 p.m., Seoul is sunny.

Figure 5: Answer generation from the result of query

3.4 SQL Generation

If a query frame is selected for a question, an SQL query statement is generated from the SQL production rule of the frame. The query is sent to the DBMS to acquire the records that match to the query. Figure 4 depicts the conversion from a natural language question to its SQL expression.

3.5 Answer Generation

Based on the result of the DBMS, a natural language answer is generated. We use a rule based answer generation method. Each query frame has an answer generation pattern for the frame. For example, DIURNALRANGE_TODAY has the following generation pattern.

```
[DIURNALRANGE_TODAY]
The diurnal temperature range of $date($1) is $2°C
```

$1 and $2 are the the first and second field value of the queried result. $date() is the function that converts a normalized date expression to a natural language expression. Figure 5 shows the answer generated from the SQL query shown in Figure 4 (More sample outputs from the QA System are presented on the Appendix) .

4 Evaluation and Limitation

We have evaluated our domain restricted QA system based on precision and recall, and investigated the limitation of the our approach to the restricted-domain QA system.

For evaluation, we've collected 50 weather questions from 10 graduate students. Precision and recall rates are 90.9 % and 75.0% respectively.

The low recall rate is due to some questions related to invalid date and topic. The system provides weather forecasts for 7 days from the querying day. But some of queries are asking for a future weather outlook which is out of range (e.g. *Will it be very hot summer this year?* or *Will it be snow on this Christmas?*). Some questions asked the information that the database doesn't contain, such as UVI (ultraviolet index).

The primary reason for the wrong answer is the failure of invalid topic rejection. It is due to the insufficient of weather-domain ontology data. For example, from the question *What is the discomfort index calculated from the today's weather?*, the keyword *discomfort index* was not extracted while *weather* was extracted, because the former was not in the ontology. So the query frame WEATHER_TODAY was misselected and the system generated the wrong answer *Seoul will be sunny on March 9th 2004*.

An error was caused by the flaw of our keyword-based query frame decision approach. For the question *Can the flight for Jeju Island take off today?*, the extracted keywords are

```
EVENT : flight take_off
DATE : 03/12/04
CITY : Jeju
```

In order to know whether the flight can take off or not, the weather information of the departure city instead of the destination city (i.e. Jeju) should be returned, but our keyword based approach failed to make an appropriate query. To solve this problem, more sophisticated semantic representation, rather than the sequence of keywords, is required for the question.

5 Related Works

In this section, we compare our system with other QA-related approaches and briefly describe the distinctive characteristics of our system. Open-domain QA systems in QA track mostly extract answers from unstructrued documents. In the contrast, our system extracts answers from semi-structured web pages, which are pre-selected by us, because our system aims to achieve high precision with the sacrifice of the coverage of questions.

Natural language front ends for databases (Copestake and Jones, 1990) and our system handle user questions similarly. However, our system has information extraction part that makes the database be updated regularly and automatically. Moreover, our system returns natural language responses to users.

The START system (Katz, 1997) is a web-based QA system. It uses World Wide Web as knowledge resource. Unstructured natural language sentences are indexed in the form of ternary expressions and stored in RDB. The START system covers much wider domain of questions than ours, however, it seems that the system returns more wrong answers than ours, because we extract the answer only from semi-structured documents.

The Jupiter system (Zue et al., 2000) is a conversational system that provides weather information over the phone. Based on the Galaxy architecture (Goddeau et al., 1994), Jupiter recognizes user question over the phone, parses the question with the TINA language understanding system (Seneff, 1992) and generates SQL and natural language answer with the GENESIS system (Baptist and Seneff, 2000). The generated answer is synthesized with the ENVOICE system. Even the Jupiter system deals with the same domain, ours can process a bit wider-range of weather topics. Our QA system can cover the question which requires inferences such as *When is the best day for washing my car in this week?* Moreover, our system has an ability of inferring missing information from the user profile and the inferring algorithm.

6 Conclusion

This paper describes the practical QA system for restricted domains. To be practically used, our system tries to achieve high precision at the sacrifice of question coverage.

To achieve high accuracy, we pre-designate semi-structured information resource webpages and extracted domain-specific information from them. We also prepare a domain-specific ontology and query frames for the question analysis. The user's request in natural language is converted into SQL expression to generate an answer for the question. Testing with a small set of queries on weather domain, the QA system showed 90.9% of precision and 75.0% of recall. By restricting the coverage of questions, our system could achieve relatively high precision. However, the figures are not enough for a real practical system.

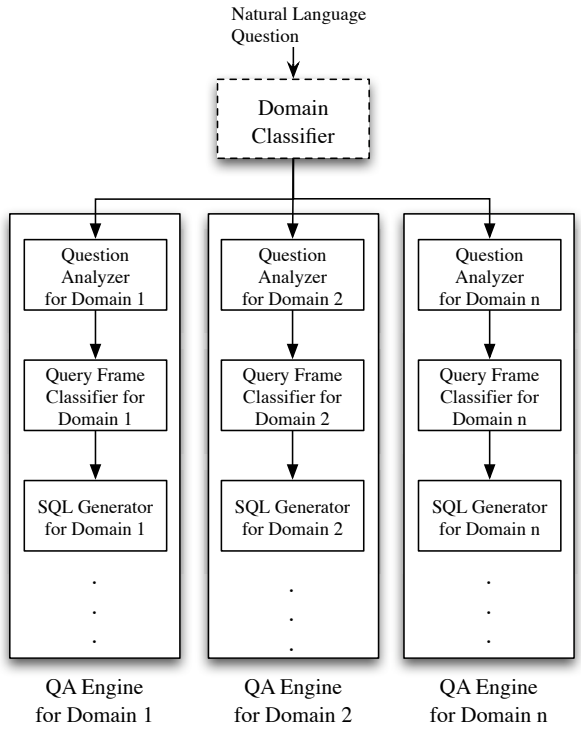

Figure 6: A domain classifier selects a proper restricted domain QA engine

Much work is left for our future work. First, we are expanding the domain for the system. A domain classifier will be added to the QA system to process multiple-domain questions, as represented in Figure 6. We will separate domain dependent resources (query frames, ontology containing domain-dependent information, and etc.) and domain independent resources (linguistic resources, and ontology for domain-independent information) to allow easier domain expansion.

Second, the information extractor has to be upgraded. Currently, the QA system is using hand-coded wrappers, and the wrappers cannot extract necessary information robustly when the webpages are modified. We are developing an information extractor that can recognize the modification of the webpages and modify the wrappers automatically. The upgraded information extractor will improve the robustness of our system.

Finally, we will increase the size of ontology to cover more question types. From the experimentation, we realize that a larger ontology for weather terms is necessary to classify a question correctly. It seems more query frames are necessary for more proper answers to the users' requests.

References

L. Baptist and S. Seneff. 2000. Genesis-II: A versatile system for language generation in conversational system applications. In *Proceedings of International Conference on Spoken Language Processing*, October.

A. Copestake and K. Sparck Jones. 1990. Natural language interfaces to databases. *The Knowledge Engineering Review*, 5(4):225–249.

C. Fellbaum. 1998. *WordNet: an Electronic Lexical Database*. The MIT Press.

D. Goddeau, E. Brill, J. Glass., C. Pao, M. Phillips, J. Polifrini, S. Seneff, and V. Zue. 1994. Galaxy: A human language interface to on-line travel information. In *Proceedings of International Conference on Spoken Language Processing*, pages 707–710.

B. Katz. 1997. Annotating the World Wide Web using natural language. In *Proceedings of the 5th RIAO conference on Computer Assisted Information Searching on the Internet*.

N. Kushmerick. 1997. *Wrapper Inductin for Information Extraction*. Ph.D. thesis, University of Washington.

M. Light, A. Ittycheriah, A. Latto, and N. MacCracken. 2003. Reuse in question answering: A preliminary study. In *New Directions in Question Answering: Papers from the 2003 AAAI Symposium*, pages 78–86.

S. Seneff. 1992. Tina: A natural language system for spoken language applications. *Computational Linguistics*, 18(1):pp. 61–86.

G. Sigletos, G. Paliouras, C. D. Spyropoulos, and M. Hatzopoulos. 2003. Mining web sites using wrapper induction, named entities and post-processing. In *1st European Web Mining Forum Workshop at ECML/PKDD-2003*.

Ellen .M. Voorhees. 2004. Overview of the TREC 2003 question answering track. In *Proceedings of the 12th Text Retrieval Conference*.

V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. J. Hazen, and L. Hetherington. 2000. Jupiter: A telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing*, pages 100–112.

Appendix: Example of QA

Q: Is it cold outside?

A: The temperature of Seoul at 1 p.m. is 11° C.

Q: Will it be cold today?

A: The highest temperature of Seoul on March 9th 2004, will be 12° C.

Q: What's the diurnal temperature range of today?

A: The diurnal temperature range of Seoul is 1°C.

Q: I am planing a picnic to Yongin on these weekends, and which day would be better for the picnic?

A: Yongin will be sunny on March 3rd 2004. Yongin will be partly cloudy on March 4th 2004.

Q: What is the ultraviolet index?

A: The system cannot answer to that topic.

Q: Can I have car wash today?

A: The precipitation probability of Seoul this afternoon is 10%.