行政院國家科學委員會專題研究計畫　期中進度報告

問答系統技術研發(2/3)－語境為本問答系統之研究

計畫類別：個別型計畫
計畫編號：NSC92-2213-E-002-026-
執行期間：92 年 08 月 01 日至 93 年 07 月 31 日
執行單位：國立臺灣大學資訊工程學系暨研究所

計畫主持人：陳信希

計畫參與人員：林川傑

報告類型：精簡報告
報告附件：出席國際會議研究心得報告及發表論文
處理方式：本計畫可公開查詢

中　華　民　國 93 年 4 月 26 日

Abstract

Answer types are very useful information in question answering. After deciding answer types, a QA system can easily select answers only from those candidates which match the answer type in the documents. This project report proposed three models to automatically find answer candidates, including Self-Evident Noun Phrases, WordNet Descendants, and Corpus Candidates. Among the testing questions, fusion of these three models proposed correct answer candidates over half of the questions. Besides, we also consider question answering in context task.

Keywords: answer fusion (答案融合), context information (語境資訊), question answering (問答系統)

1 Introduction

In recent years, question answering has become a popular research topic. Since 1999, TREC QA-Tracks (Voorhees, 2001) provided important evaluation test beds to develop question answering systems. There have been 1,893 questions, together with their correct answers found in the document set, as well as the surrounding text of the answers.

Answer type is important information used among most teams in TREC QA-Tracks. QA systems first analyze input questions and decide which types of answers are required. For example, if we know that a question is asking for a person, it would be better to report a personal name as an answer.

Because answer types cannot be enumerated completely, it is impossible to list all the answer types and design an answer candidate extractor for each type. In this project report, we propose three models to extract answer candidates automatically from the corpus based on information fusion.

2 Answer Types and Candidates

Each participating team of TREC QA-Tracks has its own answer type classification. Harabagiu *et al*. (2001) encoded 38 answer types in an ANSWER TAXONOMY. Hovy *et al*. (2001) defined 140 types in the Webclopedia project. These answer types are mostly named entities, such as persons, countries, dates, plants, *etc*. The participants have to implement an answer candidate extractor, or a named entity identifier, corresponding to their own answer type classification. Here are some examples taken from TREC QA-Track questions with their possible answer types attached at the end:

Q971: How tall is the Gateway Arch in St. Louis, MO? [LENGTH]
Q998: What county is Phoenix, AZ in? [COUNTY]
Q1228: What is the melting point of gold? [TEMPERATURE]

When the answer type of a question is decided, a QA system finds out all occurrences of terms which match this answer type, and considers them as answer candidates. The QA system will rank these candidates, and propose the most proper answer candidates will be proposed.

Answer types can be divided into two classes – say, named entities and entity sets. For named entities, we want to know the name given to a specific entity, such as "*Canada*" (a country), "*Venus*" (a planet), or "*Titanic*" (a ship), *etc*. For entity sets, what we want is a concept denoting a set of entities, such as *duck* (a kind of bird), *rose* (a kind of flower), or

dictionary (a kind of book), *etc*.

Answer candidates of the first class are often identified by named entity recognizers for the pre-classified answer types of each QA system. Candidates of the second class need some world knowledge to capture. One possible resource is WordNet, which includes the hierarchy of entities. To answer questions like "*What kind of bird can …?*", any descendant of "bird" in WordNet can be regarded as answer candidates.

In fact, not all of the questions can be classified into pre-defined answer types. In the named-entity class, there are so many entity types which can be *named* that it is not easy to define all possible named-entity sets, not to mention to design a system to identify them all. In the entity-set class, not all terms in the world are collected in WordNet (e.g., "*birthstone*" in TREC questions). Besides, the knowledge collected in WordNet (Fellbaum, 1998) is absolute hypernymy/hyponymy relationship. For example, WordNet does not provide relationship between "*habitat*" and "*mature tree*" in the following example:

Q217: What is the habitat of the chickadee?

Ans: oak tree, mature tree, meadow, …

Hyponyms of "*habitat*" in WordNet 1.7 is "*habitation*", which has two hyponyms: "*aerie, aery, eyrie, eyry*" and "*lair, den*". Maybe the information "*oak trees can be a habitat*" is collected in some knowledge base, but we do not know where it is.

3 Information Fusion

In question answering, there may exist a single piece of text which offers the information needed to answer a question. In such a case, we can extract the answer directly from the text. For example:

Q894: How far is it from Denver to Aspen?

Ans: 204 miles

Text: Aspen is 204 miles from Denver.

In the above example, this single passage explicitly mentions a DISTANCE-QUANTITY, and its end locations are Aspen and Denver, which exactly matches question Q894.

But there may not always exist sufficient information in a sentence to answer a question. A QA system may have to gather together pieces of information scattering in different documents or different pieces in a document in order to find the answer.

Information fusion is the process to handle pieces of information from different documents to answer a question. Sometimes the answer selection is decided from multiple pieces of texts. Sometimes there are more than one answer found in the corpus, but we have to decide whether these answers are exclusive, individual, or to be combined. Here are some examples that information fusion has to deal with:

(1) From multiple passages to one answer

A question can be decomposed into two or more than one sub-question. For example,

Q: Where was the first president of the United States born?

It can be decomposed into two sub-questions: "WHO was the first president of the United States", and "WHERE is his birthplace". It is possible that the answers for the first sub-question may appear in many documents while the answer for the second sub-question

appears in other documents.

(2) Contradictory answers

When different answers are reported, they may be contradictory. The most significant case is news stories for the same event reported in different time. For example,

Q: Who murdered Mary?

D_1 (in 1996): John was judged guilty for murdering Mary.

D_2 (in 1997): The police found new evident that Tom murdered Mary.

For QA systems, "*John*" and "*Tom*" are both effect answers from their surface texts. But there is only one true answer to this question, which is "*Tom*".

(3) Individual answers

Sometimes different answers are individually correct. For example,

Q378: Who is the emperor of Japanese?

The name of any previous Japanese Ten-On will be considered as a correct answer.

(4) Answers which have to be combined

There are two kinds of possible cases to combine answers. One is aggregation of quantity answers. For example,

Q: How many people were killed by cancer in Europe?

A QA system first finds out the death tolls in the European countries, and gives the sum of these numbers as the answer.

The other case is summarization of multiple passages. Questions asking for opinions, methods, status, or procedures often require longer answering passages. Texts extracted from different documents contain redundant or novel information which has to be removed or added before being reported to users.

Cases 2, 3, and 4 can be regarded as "answer fusion", because the fusion is mainly done on answer part. In Case 1, information fusion is used to resolve question terms and helps to detect correct answers in the next step.

4 Automatic Answer Candidate Selection

4.1 What-Question Type

For 5W1H questions, the targets for *who*, *where*, and *when* are clearer than those of the other three. The answer for *how*-question is non-entities, so that it is not major focus of this project report. The following only considers *what*-question and *which*-question.

There are four cases of *what*-questions:

1. "*What X VP?*" or "*N V what X?*"
   E.g. Q427: "*What culture developed the idea of potlatch?*"
   E.g.Q934: "*Material called linen is made from what plant?*"
   Answer candidates are those which are *X*'s, such as cultures or plants in this example.
2. "*What be the X-NP?*"
   E.g.Q586: "*What is the chemical symbol for nitrogen?*"
   Answer candidates are those which are *X*'s, such as chemical symbols in this example.

3. "*What*" alone as a subject or an object where its main verb is not *be*-verb
   E.g.Q552: "*What caused the Lynmouth floods?*"
   Its answer type does not directly appear in the question.
4. DIFINITION questions
   E.g. Q600: "*What is typhoid fever?*"
   Answers to such questions are definitions or descriptions.

In this project report, we experimented on only the first and the second cases. For the fourth case, i.e., DIFINITION questions, no answer candidates are needed to answer a question. Instead, gloss information or definition pattern is more helpful. For the third case, one possible way to find answer candidates is to gather all the terms as subjects (or objects) of this main verb. It remains future work and is not discussed in this project report.

For the first and the second cases, answer candidates are those which can be X's. If Y is the answer to a question Q "What X does something?", the information of "Y is an X" and "Y does something" may not appear in the same passage, even not in the same document. Information fusion is needed to gather these pieces of information together in order to answer such questions.

Our idea of answer candidate selection by information fusion is: find instances of *X* in a knowledge base; assign *Y* as one of the instances and check if "*Y* does something". If so, this instance is reported as an answer. Instances finding procedure is described in Section 5.

4.2 Question Focus

For a *which*-question or *what*-question in the first and the second cases, our system first identifies its *X* part, which is referred as "question focus" by Harabagiu *et al* (2000). We use this term but with slightly different meaning.

After syntactic parsing, if the word "*what*" or "*which*" alone is an NP, then it is in the second case and our system extracts the noun phrase after the *be*-verb as its question focus. If "*what*" or "*which*" is in a noun phrase with other words, it is in the first case and our system assigns its question focus as the noun phrase which "*what*" or "*which*" is in, but excludes the word "*what*" or "*which*".

Because it does not guarantee that we can find at least one instance of this question focus in the knowledge base, we have to relax the range of focus if necessary. Other possible foci are the head noun phrase of the question focus, and the remaining phrase with removing leading article, attaching propositional phrase, or any modifier. If the question focus is in the form of "*kind of NP*", "*type of NP*", or "*name of NP*", *etc*., possible focus is the noun phrase after "*of*".

In the following example, a question and its possible foci are demonstrated in sequence:

Q254: What is California's state bird?
Foci: California's state bird
state bird
bird

4.3 Corpus Candidates

DIFINITION Instances

In order to find instances of an entity set, we adopted DEFINITION patterns from Ravichandran

and Hovy (2002), and from Soubbotin (2001). DEFINITION questions are a special group in question answering. Such a question asks for a definition of a term, or a description of a specific person or entity.

In Ravichandran and Hovy's system, they made experiments on six question types. One of the six question types is DEFINITION. They collected pairs of questions and the corresponding answers as examples, and automatically learned their co-occurrence patterns in the knowledge base. Some example DEFINITION patterns are listed below:

<NAME> -LRB- <ANSWER> - -RRB-

<NAME> and related <ANSWER>s

<ANSWER> -LRB- <NAME> -COMMA-

in which <NAME> denotes a question term, and <ANSWER> the corresponding answer part.

Soubbotin also used DEFINITION patterns, but they made them manually. Some examples are:

<NAME> is a <ANSWER>

<ANSWER> -COMMA- <NAME> -COMMA-

<NAME> is called <ANSWER>

The reason that we use definition to find instances is: for instances of an entity set, the name of the entity set is just like the definition of the instances. Unlike the usage of these patterns in finding answers of DEFINITION questions, this time <ANSWER> part (the DEFINITION part) in the patterns is known (the entity set), and we'd like to extract <NAME> part as instances.

Syntactic information is integrated into these patterns. Since answers are mostly entities, we forced the extracted <NAME> parts to be noun phrases (NP) or quantitative phrases (QP). We extracted the minimal noun phrase if there is no other text to the left or right of the <NAME> tag.

Equivalent Instances

In some cases, the name of the entity set is not the best definition of its instance. Moreover, it may not be an appropriate definition of the instance. For example, "*oak tree*" can be an instance of "*habitat*", but the definition of "*oak tree*" is "*a deciduous tree that has acorns and lobed leaves*".

To capture such instances, we further extracted equivalent entities in the knowledge base. That is, if any form of "*A* is *B*" appeared in the corpus, than we thought *A* could be an instance of *B*, or vise versa *B* could be an instance of *A*. Again, during extraction, *A* or *B* was restricted to an NP or QP.

4.4 Answer Candidates Selection Models

We experimented on three models to find answer candidates automatically. They are:

(1) Model A: Extracting Self-Evident NPs

If an NP's head is the same as the question focus, it is regarded as an answer candidate.

E.g. QFocus: artery

AnsCand: pulmonary artery

(2) Model B: Looking for WN Descendants

If a term is a descendant of the question focus in WordNet, it is considered as an answer

candidate.

E.g. QFocus: color
AnsCand: red
WN: red, redness
=> chromatic color, spectral color…
=> color, colour, …

(3) Model C: Extracting Corpus Candidates

If a term in the corpus matches one of the DEFINITION patterns, or an equivalent relationship (*A* is *B*) is found, it is considered as an answer candidate.

E.g. QFocus: elephant
AnsCand: Loxodonta Africana
Pat: Loxodonta Africana (African elephants)

5 Experiment

5.1 Experiment Design

We used question sets provided by TREC QA-Tracks from TREC-9 to TREC-2000. We chose *what*- and *which*-questions, but dropped those which were asking persons, countries, cities, time, and quantity. This is because the answer candidates of these questions can be provided by a common named entity recognizer. After filtering out questions with no answer in TREC QA-Tracks, 251 questions were selected to do the experiment.

Question foci were half-automatically decided. We first parsed all the selected questions by ApplePie Parser, then decided its what-question type as described in Section 4.1, and extracted the focus part together with all of its sub-NPs. Human effort was introduced to check errors produced by the parser in order to focus on only answer candidate problems.

When implementing Model A, top 1,000 documents of a given question were retrieved and served as a corpus to extract self-evident noun phrases related to this question. Each noun phrase with the head the same as one of the foci of the question was collected as an answer candidates. We also tested on smaller corpus, only top 100 documents, to see the coverage.

To evaluate Model B, we used the formal answers provided by TREC QA-Tracks. For a given question, we checked if one formal answer was a descendant of the question focus in the WordNet. If so, this question was counted as "covered", because all the WordNet descendant entries were regarded as answer candidates.

The corpus of Model C was created by querying Google[1]. Each question focus was submitted as a query to Google, but forced it to only retrieve documents containing the whole phrase if the question focus had more than one word. We retrieved the first 10,000 sentences containing the question focus in the top 1,000 documents. Each sentence in the retrieved corpus was matched against the DEFINITION patterns described in Section 4.3. If matched, the noun phrase in the <NAME> part and all its sub-NPs were extracted as answer candidates. Equivalent relationship was also examined.

[1] Google: http://www.google.com/

5.2 Result

The results are listed in Table 1. Self-Evident NPs (Model A) cover 62 questions in top 100 documents, and 85 in top 1000 documents. WordNet Descendants (Model B) cover 59 questions. Google Candidates (Model C) covers 54 questions.

The fourth column lists the coverage of combined models, where "A+C" denotes he combined model of Model A and C, and so on.

Table 1. Coverage of Models (in Numbers of Questions with Correct Answer Candidates)

Self-Evident NPs (top 100)	62	A+C	113
Self-Evident NPs (A)	85	B+C	92
WordNet Descendants (B)	59	A+B	120
Google Candidates (C)	54	A+B+C	137

5.3 Discussion

Interestingly, even though self-evident NPs alone have the largest coverage, these three models in fact cover different set of questions. Therefore, they can be good complements to one another. Many descendants in WordNet do not contain the same words as their ancestors, while many self-evident noun phrases are not collected in WordNet, especially those named entities. The model of Google candidates can extract named entities or senses not self-evident and not collected in the WordNet. Some examples are listed below. Each of them was extracted in only one model.

Q254: What is California's state bird?

A: quail (WordNet Descendants)

Q261: What company sells the most greeting cards?

A: the Hallmark card company (Self-Evident NPs)

Q355: What is the most expensive car in the world?

A: Bugatti Royale (Google Candidates)

The combined model of Model A and C covers 113 questions, and the combined model of Model B and C covers 92 questions. This means that model C does improve the coverage of answer candidates comparing to the coverage of Model A or B alone. Finally, the combined model of all three models covers 137 questions, which are more than a half of the testing questions.

The result of Model B is not exactly the performance using WordNet, because the formal answers provided by TREC dropped the words already occurring in the questions. For example, the answer of Q 1256: "*What is the only artery that carries blue blood from the heart to the lungs?*" is "*pulmonary artery*", which is a descendant of "*artery*". But the given formal answer is "*pulmonary*". Even so, the missing coverage of WordNet is the set of self-evident phrases. It does not affect the coverage of combined model too much.

There are many possible reasons of the low coverage of Corpus Candidates. One is that we only match patterns in top 1,000 documents. Many extracted candidates are redundant, especially those frequent entities.

The performance of DEFINITION patterns in this experiment is not yet clear. It was often that erroneous noun phrases were extracted. For example, one of the apposition patterns, "<NAME> -COMMA- <ANSWER> -COMMA-" is often mixed with the conjunction case.

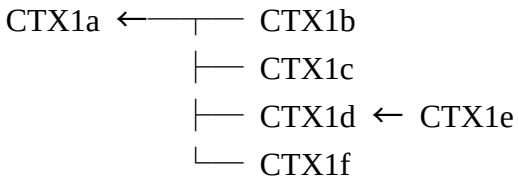
Further investigation of these patterns is necessary.

6. Context Task

Besides selection of answer candidates using information fusion, we also consider the context task. A series of questions are submitted, which are somewhat relative to the previous questions. For example, in Question CTX1:

a. Which museum in Florence was damaged by a major bomb explosion in 1993?
b. On what day did this happen?
c. Which galleries were involved?
d. How many people were killed?
e. Where were these people located?
f. How much explosive was used?

Question CTX1a asks the name of the museum. Question CTX1b continues to ask the date of the event mentioned in Question CTX1a, so this question and its answer are important keys to Question CTX1b. Question CTX1c asks more details of Question CTX1a, but irrelevant to Question CTX1b. So is Question CTX1d. But Question CTX1e refers to both Question CTX1a and CTX1d. We can draw a dependency graph of this series of questions as below:



If a question is dependent on one of its previous question, it is obvious that the information relative to this previous question is also important to the present question. Thus the system has to decide the question dependency.

We proposed a simple strategy to judge the dependency. Because the first question is the base question of this series, every subsequent question is dependent to the first one. After reading a question, if there is an anaphor or a definite noun phrase whose head noun also appears in the previous question, we postulate that this question is dependent on its previous question.

Next issue is that how we can use the dependency information in finding answers as well as its context information. After answering a single question, the system has located some answering candidates together with documents and segments of texts in which these candidates appear. Such information can be used to answer its subsequent dependent questions, as well as the keywords of the question itself. Note that context information can be transitive. In the above example, Question CTX1e consults the information that Question CTX1d itself owns, and Question CTX1d refers to, i.e., Question CTX1a.

In our experiment, we only consider the keywords and their weights as the context information. Furthermore, we assign the lower weights to the keywords in the context information so that the importance of recent keywords cannot be underestimated. The answers to the previous question remain their weights because they are new information. The question type is decided by the present question.

The accompanying issue is that how confident an answer is included in the context information. This is because we may find the wrong answers in the preceding questions and those errors may be propagated to the subsequent questions. Moreover, do these five answers

have the same weight? Or we trust the answers of the higher ranks than those of the lower ones, or only the top one is considered.

These issues are worthy of investigating, but not yet implemented in the experiment of this year. We assign weights to the previous answers according to the following equation:

$$weight(x) = weight\_NE(x) \times \sqrt{(6 - rank(x))/5} \times weight\_PreAns(x)$$

where *weight_NE*(*x*) assigns higher weight if *x* is a named entity; *rank*(*x*) is the rank of *x*, and *weight_PreAns*(*x*) is a discount to the previous answers because they may be wrong. The square root part tries to assign higher weights to the higher-ranked answers.

Because only relevant documents to the first questions are provided, and we do not implement an IR system on TREC data, we cannot do a new search when answering the subsequent questions. Our solution is to search the same relevant set of the first question.

We submitted one run this year. Its main algorithm followed the first run of the main task.

There is still no formal evaluation of this task. The MRR of all 42 question of our result is 0.139. 4 of the first questions are correctly answered. Answers of at least one of the subsequent questions can also be found in each of these 4 series. Only one of the series is fully answered.

7. Conclusion and Future Work

In this project report, we investigated the coverage of different answer candidate extraction models. We also proposed a method to extract candidates from a large corpus. The extraction was based on the idea of information fusion, and patterns were employed to detect possible candidates.

The results of our experiments show that the three models, i.e., Self-Evident Noun Phrases, WordNet Descendants, and Corpus Candidates, have their respective coverage, and they can complement one another.

In the future, the extraction patterns of Corpus Candidates should be more carefully investigated. We will also try to find out new patterns to capture those un-answered questions.

The detection of answer candidates is very time-consuming. It will be great if such detection can be done in indexing time of IR system, and the relationships between foci and candidates can be kept in the index. It is so called QA-based indexing mentioned in Hirschman and Gaizauskas (2001).

References

Christiane Fellbaum (Ed.) (1998) WordNet: An Electronic Lexical Database, The MIT Press, 1998.

Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Girju, Vasile Rus, and Paul Morarescu (2001), "The Role of Lexico-Semantic Feedback in Open-Domain Textual Question Answering," *the Proceedings of the 39th ACL and 10th EACL*, pp. 274-281, 2001.

Sanda Harabagiu, Marius Pasca, and Steve Maiorano (2000), "Experiments with Open-Domain Textual Question Answering," the Proceedings of the 18th COLING, pp. 292-298, 2001.

Lynette Hirschman and R. Gaizauskas (2001) “Natural Language Question Answering: the View from Here,” Natural Language Engineering, Cambridge University Press, Vol. 7, No. 4, 2001, pp. 275-300.

Eduard Hovy, Ulf Hermjakob, and Chin-Yew Lin (2001), “The Use of External Knowledge in Factoid QA,” the proceedings of TREC 2001, pp. 644-652, 2001.

Deepak Ravichandran and Eduard Hovy (2002), “Learning Surface Text Patterns for a Question Answering System,” *the Proceedings of ACL*, 2002.

M. M. Soubbotin (2001), “Patterns of Potential Answer Expressions as Clues to the Right Answers,” *the Proceedings of TREC 2001*, pp. 293-302, 2001.

Ellen Voorhees (2001) “Overview of the TREC 2001 Question Answering Track,” the Proceedings of TREC-10, pp. 42-51, 2001.

行政院國家科學委員會專題研究計畫

問答系統技術研發(2/3)-語境為本問答系統之研究

出席國際會議報告

計畫類別： 個別型計畫
計畫編號： NSC92－2213－E－002－026
執行期間： 92 年 8 月 1 日 至 93 年 7 月 31 日
執行單位： 國立臺灣大學資訊工程學系暨研究所

計畫主持人： 陳信希

報告類型： 完整報告
處理方式： 本計畫可公開查詢

中 華 民 國 93 年 2 月 26 日

出席「5th International Conference on Intelligent Text Processing and Computational Linguistics」報告

陳信希

國立台灣大學資訊工程學系

1. 會議經過

2004 年 Computational Linguistics and Intelligent Text Processing 是第五屆，2 月 15 日到 2 月 20 日在南韓漢城 Chung-Ang University(中央大學)舉行，會議地點是該校新工學院大樓演講廳，由校長 Dr. Myung-Soo Park 致詞揭開序幕。筆者 2 月 14 日下午搭長榮班機出發到漢城，抵達進到預先定的旅館已經是晚上 8:00。2 月 20 日聽完 Professor Ricardo Baeza-Yates 的演講隨即搭機返國，到家是晚上 11:00。

2. 會議內容

本屆國際會議共有 129 篇投稿，經論文審查後，接受包括 40 篇全文和 35 篇短文，筆者是以全文被錄取。分成計算語言學形式化、語意和對話、語法和剖析、詞彙分析、專有名詞辨識、語意分析、辭典和語料庫、雙語資源、機器翻譯、自然語言產生、人機互動應用、資訊檢索、問答系統、資訊擷取、文件分類和分群、自動摘要等課題進行論文發表。

本屆會議並邀請四位計算語言學知名學者做專題演講，謹摘錄如下：

(1) Professor Martin Kay, Stanford University

講題：Substring Alignment Using Suffix Trees

原始文件和翻譯文件間的句子對列，比較小單位的成分(如詞彙和片語)

對列容易被了解。這場演講提出以 suffix tree 這項資料結構解決子字串對列的問題，文件中重複的子字串出現夠多次，就可由 suffix tree 識別出。

(2) Professor Philip Resnik, University of Maryland

講題：Exploiting Hidden Meanings: Using Bilingual Text for Monolingual Annotation

supervised learning 通常需要搭配大量標記過的資料，但是標記資料需要考量人力投入的負擔。因此，又有研究人員強調以 unsupervised learning 的方式，以少量標記資料，或甚至於不需要標記資料，就可達到目標。本演講談到如何運用大量句子間的平行對譯，來掌握隱藏的意義，並藉以標記單語語料庫，最後以詞彙意義分析(WSD)說明。

(3) Dr. Nick Campbell, ATR Human Information Science Laboratories

講題：Specifying Affect and Emotion for Expressive Speech Synthesis

語音合成不一定是單純的文件轉語音，本次演講談如何由說話者、語言、說話的風格、和內容資訊等的組合來產生語音。

(4) Professor Ricardo Baeza-Yates, University of Chile

講題：Challenges in the Interaction of Information Retrieval and Natural Language Processing

本次演講談論自然語言處理技術，如何與資訊檢索整合。首先以兩個領域重疊的應用講起，包括自動摘要、資訊擷取、和問答系統。接著探討可能挑戰性的應用，包括上下文的確定、語意檢索、和語意網際網路的支援。

筆者的論文”Extracting Domain Knowledge for Dialogue Model Adaptation”，是和所指導的碩士生林桂光同學共同撰寫，被安排在 2 月 16 日

下午的「語意和對話」場次中發表。這篇論文主要探討對話管理中領域轉移的問題，如何以輔助使用者的方式自動擷取領域知識。這包括對話中基本成份的擷取和分類，speech act 的掌握，和對話狀態的改變。我們以四種不同型態的語料庫來模擬知識擷取的過程，以彰顯對領域的調適能力。報告後 Professor Resnik 問起為何不用高階馬可夫模型來分析，當然主要的原因是對話語料庫的收集需要發很多的費用，目前語料庫的規模都不大。國內學者其實已經意識到對話語料庫是對話機制研究不可或缺的重要資源，目前透過中華民國計算語言學學會，和國科會計畫的協助，共同製作資源，以利國內這個領域的發展。

3. 攜回資料

會議論文集收錄於 Springer Verlag 出版的 Lecture Note in Computer Science 中，編號是 LNCS 2945。此外，也攜回會議手冊，記錄邀請講席，和論文報告者的投影片資料。

(1) Alexander Gelbukh (Ed.) Computational Linguistics and Intelligent Text Processing, Lecture Notes in Computer Science, LNCS 2945, Springer Verlag, 2004.

(2) Sang Yong Han and Alexander Gelbukh (Eds.) Handbook of Fifth International Conference on Intelligent Text Processing and Computational Linguistics, Chung-Ang University, Seoul, Korea, 15-21 February, 2004.