UAlbany's ILQUA at TREC2007

Min Wu, Chen Song, Yu Zhan, and Tomek Strzalkowski[†]
ILS, University at Albany SUNY
1400 Washington Ave. Albany, NY, USA
{minwu,tomek}@cs.albany.edu

1 Overview

TREC2007 QA track introduced a combined collection of 175GB BLOG data and 2.5GB newswire data. This introduced an additional challenge for an automatic QA system to processes data in different formats without sacrificing the accuracy. In ILQUA we added a data preprocessing component to filter out noisy blog data.

ILQUA has been built as an IE-driven QA system; it extracts answers from documents annotated with named entity tags. Answer extraction methods applied are surface text pattern matching, n-gram proximity search and syntactic dependency matching. The answer patterns used in ILQUA are automatically summarized by a supervised learning system and represented in form of regular expressions which contain multiple question terms. In addition to surface text pattern matching, we also adopt N-gram proximity search and syntactic dependency matching. N-grams of question terms are matched around every named entity in the candidate passages and a list of named entities are extracted as answer candidate. These named entities then go through a multi-level syntactic dependency matching component until a final answer is chosen. This year, we modified the component that tackles "Other" questions and applied different method in the two runs we submitted. One method utilized representative words and syntactic patterns, while the other method utilized representative words from TREC data and web data.

Figure 1 gives an illustration of components, data flow and control flow of ILQUA. The following sections give detailed discussion of each component of the system, evaluation results, conclusion and future work.

[†] Also affiliated with the Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland

2 Data Preprocessing

2.1 BLOG Data Cleaning

The TREC2007 QA BLOG data contain three types of files: permalinks, RSS feeds and blog homepages. Indexing, searching and analysis are performed on permalink documents collection because permalinks contain the actual content of the posts. As is well known, blog-data can be often noisy and messy due to the diversity of the post sources and many formats used; therefore, some cleaning and filtering is necessary in order to obtain a reduced representation, which is much smaller in volume, yet maintains the overall integrity of the original data.

Data cleaning is performed on two levels: document-level cleaning and corpus-level cleaning. At the document-level cleaning markup tags such as HTML tags, XML tags and stylesheets which don't contribute to the actual content at all, are removed. We used HTMLParser to perform this task. At the corpus-level cleaning non-English BLOG documents are removed from the corpus. We used LUCENE built-in language analyzer to classify BLOG pages into two sets (English and non-English), and then removed the non-English set from the corpus.

After the cleaning and filtering, the BLOG data is reduced to approximately 13 GBytes (or less that 10% of the original size).

2.2 Named Entity Tagging

As we mentioned above, ILQUA is an IE-driven QA system and answer extraction is performed on named entity tagged text snippets. The data corpus is first tagged with NE tagger (we use BBN's IdentiFinder). It was a very time-consuming task to annotate the whole 15 GB TREC2007 data corpus.

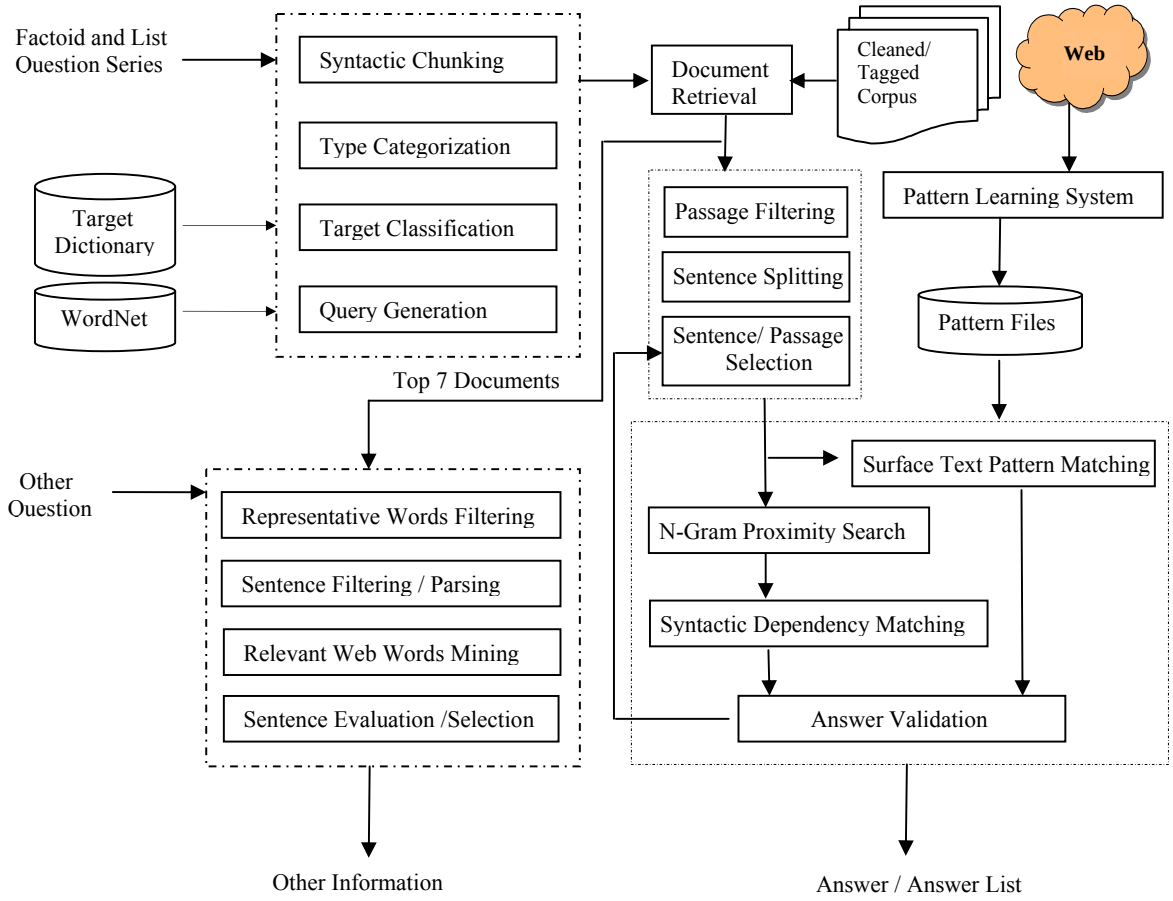

Figure 1. ILQUA Architecture

3 Question Analysis

Question analysis component consists of four modules: syntactic chunking, answer target classification, question type categorization and query generation.

Syntactic chunking splits question into a list of question terms with syntactic tags. For example, the question "When were the first postage stamps issued in the United States" will be chunked with the syntactic structure of "When_Be_NP_VP_NP". However, some questions with special answer patterns do not follow thispattern, e.g., "Born_When", "Born_Where", "Die_When", "Die_Where", "Abbreviation" etc.

The answer targets for questions are classified into named entity types that are expected in the answer. The number of named entity types that ILQUA can process is 27. The following lists all the named entity types that our system can currently process.

ANIMAL / CONTACT_INFO / DISEASE
EVENT / FAC / GAME / GPE / LANGUAGE
LAW / LOCATION / NATIONALITY
ORGANIZATION / PERSON / PLANT PRODUCT /
SUBSTANCE / WORK_OF_ART
CARDINAL / MONEY / ORDINAL / PERCENT
"QUANTITY / DATE / TIME
COLOR / GENRE / OCCUPATION

ILQUA classifies answer target with the aid of pre-defined rules and dictionary. For example, if questions begin with "When", "Where" and "Who", the answer targets are simply assigned as "Date", "Location" and "Person"; if questions begin with pattern "How+Adj.", answer targets are assigned according to the adjectives; if questions begin with "What_Be", "What_NP", "Which_Be", "Which_NP", the key term of noun phrase is mapped to appropriate answer target type. We built a dictionary with around 8000 entries to map noun

phrase patterns to named entity types which can be processed by ILQUA. The mapped named entity type is set as the major answer target type and the specific noun phrase is set as the minor answer target type. For example, if the major answer target is "Quantity", the minor answer target could be "age", "distance", "height", "speed" etc. This two-level answer target categorization is helpful to answer validation.

Query expansion is done with the aid of WordNet to find the morphological forms and synonyms of verbs. We didn't use the noun synonyms to expand the query because the noise introduced by some noun synonyms would reduce the retrieval precision.

4 Passage Retrieval and Filtering

We used two different IR engines this year. For BLOG document indexing and searching, we used Lucene. For newswire data indexing and searching, we used Inquery (by University of Massachusetts at Amherst). The top 100 BLOG documents retrieved by Lucene and the top 100 newswire documents retrieved by Inquery are segmented into passages. Then these passages are filtered by answer target type. Passages without named entity of answer target type are filtered out. All the NE tags in the passages except the tags of answer target are filtered out for later processing. The remaining passages are again filtered by question terms and topic terms.

5 Answer Extraction

5.1 Surface Text Pattern Matching

Surface text pattern matching has been adopted by some researchers (Ravichandran & Hovy 2002, Soubbotin 2002) in building QA system during the last few years. Although surface text pattern matching is a simple method, it is very effective and accurate to answering specific types of questions.

Patterns used in ILQUA are represented as regular expressions with terms of "NP", "VP", "VPN", "ADVP", "be", "in", "of", "on", "by", "at", "which", "when", "where", "who", ",", "-", "(" etc. Some questions contains more than one noun phrase, we number these noun phrases according to their orders in the questions. The following regular expression list is a sample of answer patterns to question type "when_do_np1_vp_np2".

```
NP1 VP NP2 in <Date>([^<>]+?)<\/Date>
NP1 VP NP2 on <Date>([^<>]+?)<\/Date>
NP1.{1,15}VP NP2 in <Date>([^<>]+?)<\/Date>
NP2 in <Date>([^<>]+?)<\/Date>.{1,15}NP1
<Date>([^<>]+?)<\/Date> NP1 VP.{1,15}NP2
NP1.{1,15}VP.{1,30} on <Date>([^<>]+?)<\/Date>
NP1.{1,15}NP2 on <Date>([^<>]+?)<\/Date>
NP1.{1,30}NP2 on <Date>([^<>]+?)<\/Date>
<Date>([^<>]+?)<\/Date>.{1,15}NP1.{1,50}VP
NP1's NP2 in <Date>([^<>]+?)<\/Date>
<Date>([^<>]+?)<\/Date>.{1,15}NP1 VP NP2
```

These patterns were automatically mined from web and organized by question type. We used previous TREC QA questions and answers as sample question-answer pairs. The details of answer pattern mining process have been explained in our previous TREC QA reports.

When applying these patterns to specific question, the terms such as "NP", "VP", "VPN", "ADVP" and "be" should be replaced with the corresponding question terms. The replaced patterns can be matched directly to the candidate passages and answer candidate be extracted quickly with Java tools. The number of patterns varies by specific question type. Some question type has up to several hundred patterns. Only patterns with score greater than some empirically determined threshold are applied in pattern matching.

5.2 N-gram Syntactic Dependency Matching

Proximity search as an IR method has been used in QA before (Han, Chung and Kim 2004). We applied a combined method of n-gram proximity search and syntactic dependency matching to process questions whose answer cannot be extracted by surface text pattern matching.

Around every named entity in the candidate passages, question terms as well as topic terms are matched as n-grams. Each term is tokenized by word. We matched the longest possible sequence of tokenized word within the 100-word sliding window around the named entity. Once a sequence is matched, the corresponding tokens are removed from the token list and the same searching and matching is repeated until the token list is empty or

no sequence can be matched. The candidate named entity is scored by the average weighted distance score of matched question terms and topic terms.

Let $Num(t_i...t_j)$ denote the number of all matched n-grams, $d(E, t_i...t_j)$ denote the word distance between the named entity and the matched n-gram, $W1(t_i...t_j)$ denote the topic weight of the matched n-gram, and $W2(t_i...t_j)$ denote the length weight of the matched n-gram. If $t_i...t_j$ contains topic terms or question verb phrase, 0.5 is assigned to *W1,* otherwise 1.0 is assigned to *W1*. The value assigned to length weight *W2* is determined by λ, the ratio value of matched n-gram length to question term length. How to assign the value of *W2* is illustrated as follows.

$$W2(t_i...t_j)=0.4 \quad if\ \lambda<0.4$$
$$W2(t_i...t_j)=0.6 \quad if\ 0.4 \le \lambda<0.6$$
$$W2(t_i...t_j)=0.8 \quad if\ \lambda>0.6$$
$$W2(t_i...t_j)=0.9 \quad if\ \lambda<0.75$$

The weighted distance score $D(E,QTerm)$ of the question term and the final score $S(E)$ of the named entity are calculated as follows.

$$D(E,QTerm)=\frac{\sum_{t_i...t_j}\frac{d(E,t_i...t_j)\times W1(t_i...t_j)}{W2(t_i...t_j)}}{Num(t_i...t_j)}$$

$$S(E)=\frac{\sum_{i}^{N} D(E,QTerm_i)}{N}$$

After the n-gram proximity search generates a list of named entities as answer candidates, the syntactic dependency matching component takes the top 20 ones as input and set the final answer.

Here we use a simple example to illustrate how syntactic dependency works. Figure 5 shows the top 5 answer candidates (underlined dates) of question "When was the first Burger King restaurant opened?" after the n-gram proximity search. We used MINIPAR (DeKang Lin) to parse the question and the sentences containing the answer candidates. Then, we matched the syntactic relation triples of the question one by one against the triples of parsed sentences. With the aid of dependency-based word similarity list (developed by DeKang Lin), we can match synonyms or highly related words between question and candidate sentence. To improve the matching accuracy, we introduced the forward matching propagation and the backward matching propagation. If there are two syntactic relations A:R1:B and B:R2:C in the question, suppose A:R:B is not matched against any relations in the parsed sentence, the forward propagation will consider the relation A:R1:C or A:R2:C. Suppose A:R1:B is matched with the parsed sentence and B:R2:C is not matched with any relations in the parsed sentences, the matching score of A:R1:B will be adjusted according to the rule of backward propagation.

As for the example question mentioned above, the parsed dependency relation triples are listed as follows:

When was the first Burger King restaurant opened?

1. The number of ***Burger King*** fast-food ***restaurant***s have reached 100 throughout Turkey since ***first*** was ***opened*** in **1995**, reported the Anatolia News Agency on Sunday.
2. Coke has supplied Burger King for most of the ***restaurant*** chain's history, starting with ***the first Burger King*** that ***opened*** in Miami in **1954**.
3. ``The company chose an available Pillsbury pancake mix brand name, Hungry Jack's, in its place. . . . ***Burger King opened*** its ***first*** four company-owned ***restaurant***s (under the Burger King name) in Sydney, New South Wales . . . in **December 1997**."
4. why some BK look-alike restaurants in Australia are named Hungry Jack's while others bear the Burger King moniker. ``The Burger King brand name was not available for use by ***Burger King*** Corp. in **1971**," BK says. ``The company chose an available Pillsbury pancake mix brand name, Hungry Jack's, in its place. . . . Burger King ***opened*** its ***first*** four company-owned ***restaurant***s .....
5. When the recall was ***first*** announced **Dec. 27**, ***Burger King*** placed an ad in USA Today, posted signs in its ***restaurant***s and sent out notices to 56,000 pediatricians.

Figure 5 Answer Candidates

~	Q:wha:A	when
~	Q:head:YNQ	~
~	YNQ:inv-be:be	be
~	YNQ:head:V	open
open	V:s:N	restaurant
restaurant	N:det:Det	the
restaurant	N:post:PostDet	first
restaurant	N:nn:N	Burger King
Burger King	N:lex-mod:U	Burger

There are two main syntactic dependency relations: the first relation is about "when" and "open" and the second relation is about "open" and "the first Burger King restaurant". These two relations are then matched with the relations in parsed sentences in the Figure 5. In the first round of syntactic matching, answer candidates "1995", "1954" and "December 1997" are matched. In the second round of matching, answer candidate "1954" get higher score because "the first Burger King" is more close to the question term "the first Burger King restaurant". So the final answer will be "1954".

5.3 Answer Validation

The goal of answer validation is to make sure the answer is returned as correct format and the answer matches both the major answer target type and minor answer target type. For example, for questions asking for "What year", the answer will be returned as year format; for questions asking for "distance", the answer should be a quantity value and contain terms such as "miles", "meters" etc.; for questions asking for "how much money", the answer should be a quantity value and contain terms of currency names such as "dollar", "pound" etc.

6 "Other" Questions

In previous TREC QA, ILQUA used syntactic patterns with semantic features to extract answer nuggets to definition ("Other") questions. This year, we integrated relative words analysis in order to improve the precision of the answers.

Firstly, candidate sentences selection is very necessary to reduce the huge volume of relevant information about each topic. We set a threshold of 100 sentences. We utilized the relevant passages retrieved from previous factoid and list questions. For each factoid/list question, we collect passages from the top 30 relevant newswire documents and top 30 relevant BLOG documents. These passages are split into sentences and sentences are thus scored by their relevance to topic. Among the final ranked sentence list, the top 100 ones are saved as later use.

Next we retrieved representative words from the top 100 sentences. In each sentence, we set a 60-word sliding window around topic phrases and select the most frequently occurring words among that frame. In addition, we also collect representative words from the web.

The candidate sentences are parsed and the parse trees are traversed bottom-up. The content at each level in the parsed tree are evaluated and assigned a score according to the following formula:

$$S^* = \sum \frac{(S_{topic} + S_{digit} + S_{rep} + S_{adj})}{4}$$

$$S = \frac{S^*}{L_{pattern} + 1} * \alpha + (1 - \alpha) * \frac{L_{opt}}{L_{content}}$$

Here S* is the sum score of every syntactic unit at each parsed tree level. For example, if at some level the syntactic parsed tree pattern is "NP JJS NN NNS", then S* is the sum of scores for "NP", "JJS", "NN" and "NNS". To be more specific, S_{topic} is calculated if a topic exists in the text content of the syntactic unit; S_{digit} is calculated if digits are contained in the text content; S_{rep} is calculated if a representative word occurred in text; and S_{adj} is calculated if there are adjective phrases present in text. $L_{pattern}$ is the number of syntactic units and $L_{content}$ is the number of words in text snippets. L_{opt} is a constant value of 64 and α is an optional value that is adjusted by experiments.

All the scored nuggets are sorted and ranked. Finally we chose the top 30 nuggets as the answers.

7 Experiments and Evaluation Results

Table 1 shows the evaluation results of two submitted runs. The factoid accuracy of ILQUA1 and ILQUA2 are the same. The F-scores for list questions in ILQUA1 and ILQUA2 differ a little because the two runs use different thresholds to control number of returned answers. The pyramid scores for other questions in ILQUA1 and ILQUA2 differ in the number of representative words.

Table 1 ILQUA Evaluation Result

RUN ID	Factoid	List	Other	Average Per Series
ILQUA1	0.222	0.147	0.242	0.203
ILQUA2	0.222	0.144	0.216	0.193
Best	0.706	0.479	0.329	0.484
Median	0.131	0.085	0.118	0.108

8 Conclusion and Future Work

ILQUA is an IE-driven QA system and has participated TREC QA task since 2003. It kept a stable performance in answering factoid, list and other questions from a data collection with mixed formats.

To improve the system performance, future development could include: increasing and refining the named entity categories; integrating semantic similarity into the proximity search; improving temporal context analysis techniques.

Acknowledgements

The development of ILQUA and our more advanced interactive QA system (HITIQA) is supported by the Disruptive Technology Office's Advanced Question Answering for Intelligence (AQUAINT) program.

References

Chu-Carrol, J., J. Prager, C. Welty, K. Czuba and D. Ferrucci. "A Multi-Strategy and Multi-Source Approach to Question Answering", In Proceedings of the 11th TREC, 2003.

Cui, H., K. Li, R. Sun, T.-S. Chua and M.-Y. Kan. **"**National University of Singapore at the TREC 13 Question Answering Main Task". In Proceedings of the 13th TREC, 2005.

Echihabi, A., U.Hermjakob, E. Hovy, D. Marcu, E. Melz and D. Ravichandran. "Multiple-Engine Question Answering in TextMap". In Proceedings of the 12th TREC, 2004.

Han, K.-S., H. Chung, S.-B. Kim, Y.-I. Song, J.-Y. Lee, and H.-C. Rim. "Korea University Question Answering System at TREC 2004". In Proceedings of the 13th TREC, 2005.

Harabagiu, S., D. Moldovan, C. Clark, M. Bowden, J. Williams and J. Bensley. "Answer Mining by Combining Extraction Techniques with Abductive Reasoning". In Proceedings of 12th TREC, 2004.

Hovy, E. L. Gerber, U. Hermjakob, M. Junk and C.-Y. Lin. "Question Answering in Webclopedia". In Proceedings of the 9th TREC, 2001.

Katz, B. and J. Lin. "Selectively Using Relations to Improve Precision in Question Answering". In Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering. April 2003.

Moldovan, D. and V. Rus. "Logical Form Transformation of WordNet and its Applicability to Question Answering". In Proceedings of the ACL, 2001.

Prager, J., E. Brown, A. Coden and D. Radev. "Question-Answering by Predictive Annotation". In Proceedings of SIGIR 2000, pp. 184-191. 2000.

Ravichandran, D. and E. Hovy. "Learning Surface Text Patterns for a Question Answering System". In Proceedings of 40th ACL. 2002.

Soubbotin, M. and S. Soubbotin. "Patterns of Potential Answer Expressions as Clues to the Right Answers". In Proceedings of 11th TREC. 2003.

Voorhees, E. "Using Question Series to Evaluate Question Answering System Effectiveness". In Proceedings of HLT 2005. 2005.

Wu, M., M. Duan, S. Small, T. Strzalkowski, ILQUA – An IE-Driven Questioning Answering System. In TREC-14 Proceedings. 2005.

Wu, M. and T. Strzalkowski, Utilizing Co-occurrence of Answers in Question Answering. In Proceedings of COLING-ACL 2006. July 2006.