Question Answering Approach Using a WordNet-based Answer Type Taxonomy

Seung-Hoon Na, In-Su Kang, Sang-Yool Lee, Jong-Hyeok Lee

Department of Computer Science and Engineering, Electrical and Computer Engineering Division
Pohang University of Science and Technology (POSTECH)
and
Advanced Information Technology Research Center (AITrc)

1. Introduction

In question answering (QA), answer types are semantic categories that questions require. An answer type taxonomy (ATT) is a collection of these answer types. ATT may heavily affect the performance of QA systems, because its broadness and granularity provides coverage and specificity of answer types. Cardie [1] used 13 categories for entity classification, and obtained large performance improvement, compared with the method using no categories. Also, according to Pasca et al. [3], the more categories a system uses, the better performance the system shows. For example, consider two answer type taxonomies, A={PERSON}, and B={PRESIDENT, ENGINEER, SINGER, PERSON}. Given a question "*Who was the president of Vichy France?*", we know that the more specific answer type of this question is not PERSON, but PRESIDENT. Thus, if we use ATT B, a set of candidate answers from documents can be reduced to a set of PRESIDENT entities, by excluding the other PERSON entities such as ENGINEER and SINGER. This is not the case with ATT A. However, since ATT A cannot distinguish hypernyms of PERSON, the QA system should consider much more candidate answers.

Thus far, most QA systems rely on small-scale ATTs, with the number of semantic categories ranging from 20 to 100. Normally, these ATTs are created from a beginning set of frequently-asked answer types like person, organization, location, number, etc., and then they are incrementally extended to include unexpected answer types from new questions. However, these ad-hoc ATTs may raise the following problems in QA. First, it is nontrivial to manually enlarge a small ATT to a large one, as new answer types appear. Second, ad-hoc ATTs do not allow easy adaptation for processing questions asking new answer types. For such questions, the system needs to modify an existing IE module to classify entities into new answer types. Third, previous ATTs do not have sufficient broadness and granularity, where they are expected characteristics of ATT for open-domain QA.

Therefore, at this year's TREC, we have taken a question answering approach that uses WordNet itself as ATT. In other words, our QA system maps an answer type into a concept node called a synset in WordNet. WordNet provides sufficient diversity and density to distinguish specific answer types for most questions. By using such an ontological taxonomy, we do not have the above problems with small ad-hoc ATTs.

This paper is organized as follows. Section 2 describes each module of our QA system, and section 3 shows TREC-11 evaluation results, and concluding remarks are given in section 4.

2. System Description

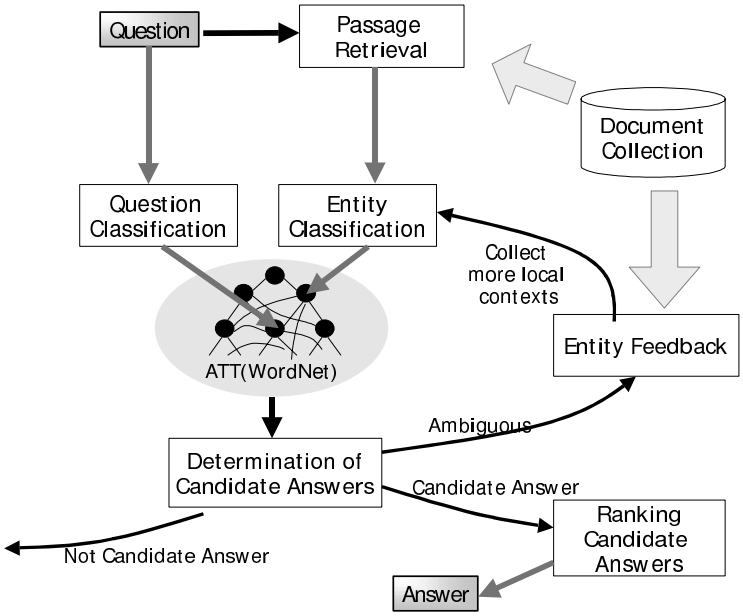


Figure 1. System Architecture

Our QA system consists of components as illustrated in figure 1. *Question Classification* analyzes a question to determine its answer type which maps to a synset in WordNet. *Passage Retrieval* formulates a vector query from the question, and retrieves relevant passages. *Entity Classification* processes top N passages from *Passage Retrieval* to classify each entity into its semantic category which as well corresponds to a synset in WordNet. A final answer is obtained from Answer Extraction by processing two sub sequent steps: *Determination of Candidate Answers* and *Ranking Candidate Answers*. *Determination of Candidate Answers* semantically recognizes all candidate answers in relevant passages by matching a taxonomic relation between the answer type of a question and the entity type of each entity, and regard matched entities as candidate answers. In the matching process, if any entity type is a hypernym of the answer type, since the entity does not have sufficient evidence to be selected as a candidate answer, thus in the case, *Entity Feedback* is invoked to collect clue words indicating more specific type from whole document collection. After *Entity Feedback*, the entity can be classified to more specific semantic category by *Entity Classification*. Unfortunately, in TREC-11, we did not perform experiment on the *Entity Feedback*. *Ranking of Candidate Answers* ranks candidate answers by calculating the similarity between a question and the passage containing each candidate answer, and generates a top candidate answer.

2.1 Question Classification

For each question, *Question Classification* determines its answer type using a set of 20 question pattern rules that were

created from analyses of previous TREC questions. First, a question is tagged with part-of-speech tags using Tree-Tagger [6]. Next, our NP-chunker performs NP chunking and marks a linguistic head of a noun phrase. The NP-chunker was designed by a simple rule-based scheme. After chunking, question pattern rules are applied to the question to determine an answer type of a question.

Lexical constraint	Semantic constraint	Answer type	Example
What (be) NP	{head of NP} <= ENTITY_{synset}	{head of NP}	*What is the president of U.S. ?*
How many NP	{head of NP} < UNIT_{synset}	{head of NP}	*How many miles ...?*
How JJ	$\exists Y$ *Value_of* ({JJ}, X) and *Unit_of* (X,Y)	*Y*	*How fast does a cheetah run?*
who		PERSON_{synset}	*Who was the lead singer for the Commodores?*

Table 1. Question Pattern Rules

Table 1 shows some examples of question pattern rules, where lexical and semantic constraints describe preconditions to be met before the question pattern is matched against a question. {*X*} indicates a WordNet synset of a lexical word corresponding to *X*. Since there are several senses for a lexical word, {*X*} is currently assumed to be the most frequent sense. '{*X*} < {*Y*}' means that {*X*} be a hyponym of {*Y*}. For example, suppose a question "*What is the governor of Colorado?"*. After performing np-chunking for this question, the question is converted into "*What is* $[\text{the governor}]_{NP}$ *of* $[\text{Colorado}]_{NP}$*?"*. This is matched with the first rule in table 1, because it satisfies a lexical constraint '*What (be) NP*', and a synset of the *NP* head GOVERNOR_{synset}, is a hyponym of ENTITY_{synset}. In this example, therefore, the answer type is a synset GOVERNOR_{synset} in WordNet.

On the other hand, to deal with questions starting with 'how', we defined an additional semantic relation *Unit_of* into WordNet as follows.

$$\text{DISTANCE}_{synset} \text{ — } Unit_of \rightarrow \text{LINEAR_UNIT}_{synset}$$

$$\text{FINANTIAL_CONDITION}_{synset} \text{ — } Unit_of \rightarrow \text{MONETARY_UNIT}_{synset}$$

This relation is also used in question pattern rules. For example, given a question "*How far would you run if you participate in a marathon?*", it satisfies the lexical constraint of the third pattern in table 1, where *JJ* is a part-of-speech tag for an adjective. Because a synset FAR_{synset} for 'far' has a *Value_of* relation with a synset DISTANCE_{synset}, the variable *X* in the semantic constraint becomes DISTANCE_{synset}. Since, we defined that DISTANCE_{synset} has a *Unit_of* relation with $\text{LINEAR_UNIT}_{synset}$ in WordNet, the variable *Y* becomes $\text{LINEAR_UNIT}_{synset}$. Therefore, the answer type of the question is set to $\text{LINEAR_UNIT}_{synset}$.

2.2 Passage Retrieval

Passage Retrieval consists of two processing steps: document retrieval and passage ranking. For document retrieval, we have developed an IR system based on a vector space model. To extract terms, we use a stop word list provided by the SMART system [5], from which 30 words such as *first*, *second* are eliminated because they provide important clues in QA. We do not use stemming, but use Tree-Tagger [6] to obtain root forms which are used as terms. To weight each term, we employ the weighting formula *nxx.bfx* introduced by Salton and Buckley [4].

From top 1000 documents generated by a document retrieval module, passage ranking identifies all legitimate passages in each document and rank them, using cover density ranking by Clarke et al. [2]. A passage unit is the minimal number of subsequent sentence including maximal number of query terms. No three passages are taken from the same document and no passages can contain more than five sentences.

2.3 Entity Classification

From the top 15 passages obtained by passage retrieval, *Entity Classification* classifies each entity occurring in the passages into a semantic category that maps to a synset in WordNet. First, we access entity databases. There are three kinds of entity databases: person names, location names, and organization names. If an entity is found in one of these databases, then the type of the entity is set to the type of the entity database. Otherwise, the entity is looked up in WordNet, and its corresponding synset is selected as the entity type. If the entity is not found even in WordNet, a clue word related to the entity is searched within the document containing the entity. Here, a clue word means a word that can indicate a semantic category of the entity. To recognize a clue word, we use a set of 30 type indicator patterns, which was empirically constructed. Table 2 shows some examples of type indicators. For instance, appositive constructions may provide type words for entities. For entities like "*Nancy Powers*" and "*Thomas*" in table 2, we know that their entity types are $DIRECTOR_{synset}$ and $DETECTIVE_{synset}$ respectively, since the entity and its type word are grouped into an appositive construction.

Type indicator	Examples
Appositive	1) **Nancy Powers**, a sales director for a medical company 2) **Steve Thomas**, the public detective
Role	1) Mrs. **Aquino,** 2) **New York** city
Relative clause	1) **Kenneth Starr**, who
Preposition	1) in **Washington**
Unit	1) **8,000** miles, 2) **8** miles per hour

Table 2. Examples of Type Indicators

2.4 Determination of Candidate Answers

Determining whether an entity is a candidate answer or not is based on taxonomic relations in WordNet between an entity type and the answer type of a question. Figure 2 shows three possible taxonomic relationships between the entity type and the answer type.

t_q : Answer type t_e : Entity type

A > B : A is a hypernym of B

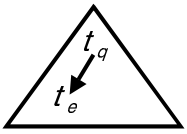
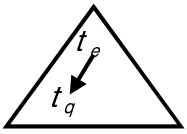
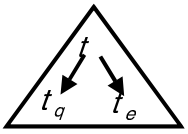
Taxonomic Relationship	1) $t_q \geq t_e$	2) $t_q < t_e$	3) $t > t_e, t > t_q$
			
Candidate Answer?	Completely Candidate Answer	Ambiguous: Need more local contexts	Not Candidate Answer

Figure 2. Taxonomic Relationships between an Answer Type and an Entity Type

The first is the case where the answer type is a hypernym of an entity type. The second is the case where an entity type is a hypernym of the answer type. The third includes all the other cases. As an example of the first case, suppose that the answer type is $PERSON_{synset}$ and an entity type is $ENGINEER_{synset}$. Then, since $PERSON_{synset}$ is a hypernym of $ENGINEER_{synset}$, the entity becomes a candidate answer. As an example of the third case, suppose that the answer type is $ENGINEER_{synset}$ and an entity type is $PRESIDENT_{synset}$. In this example, because the entity is not an $ENGINEER_{synset}$, it cannot become a candidate answer. As an example of the second case, if the answer type is $PRESIDENT_{synset}$ and an entity type is $PERSON_{synset}$, it is not clear whether the given entity is a $PRESIDENT_{synset}$ or not. So, candidate answer determination fails. In this case, an entity feedback module is fired to acquire another clue words from the whole document collection, which may contain the more specific semantic category for the same entity.

2.5 Ranking of Candidate Answers

After recognizing candidate answers, each candidate answer is ranked by the formula (1).

$$Score = Type_score \times Context_score$$

$$Context_score = \sum_{(t_i, t_j) \in T} w_{q,(t_i,t_j)} \times w_{d,(t_i,t_j)} \quad \text{---------- (1)}$$

$$T = Second\ order\ term\ set$$

In formula (1), *Type_score* is determined by WordNet taxonomic relationships between the answer type of the question and the entity type of the candidate answer. When the relationship belongs to the first case in figure 2, its *Type_score* is 1, and in

the second case, *Type_score* set to 1/2 since currently our QA system does not support entity feedback module.

Context_score is a score calculated from proximity between the candidate answer and the question words in the passage where the candidate answer appears. To compute a context score, first we convert the passage into a second-order vector (SOV) that is different from the traditional first-order vector (FOV). The difference is that a term in FOV consists of a single lexical word and a term in SOV consists of two lexical words. For example, when a vocabulary set is V ={A, B, C}, the possible term set in SOV is T = {AB, AC, BC}, where each element is called by a second-order term. To weight a second-order term, we assume a proximity hypothesis that the closer two participating lexical words of the second-order term is in a passage, the more important the term is. According to this hypothesis, given any two lexical words t_i and t_j in second order term set, we calculate the weight of the second-order term (t_i, t_j) by formula (2), where $sent_dist_{d,(t_i,t_j)}$ and $pos_dist_{d,(t_i,t_j)}$ is a sentence distance and a positional distance between t_i and t_j on the passage d respectively.

$$w_{d,(t_i,t_j)} = proximity_{d,(t_i,t_j)} \quad \text{---------- (2)}$$

$$proxmity_{d,(t_i,t_j)} = \begin{cases} \dfrac{1}{sent_dist_{d,(t_i,t_j)} \times \log(pos_dist_{d,(t_i,t_j)} + 1)} & \text{if } t_i \text{ and } t_j \text{ exists in the passage } d \\ 0 & \text{otherwise} \end{cases}$$

For the question *q*, we use formula (3), where $pos_dist_{q,(t_i,t_j)}$ is a positional distance between t_i and t_j on the question q.

$$w_{q,(t_i,t_j)} = \left(\frac{idf_{t_i} + idf_{t_j}}{2} \right) \times proximity_{q,(t_i,t_j)} \quad \text{---------- (3)}$$

$$proximity_{q,(t_i,t_j)} = \begin{cases} 1 & \text{if } t_i \text{ and } t_j \text{ exists in the same noun phrase within the question } q \\ \dfrac{1}{pos_dist_{q,(t_i,t_j)}} & \text{if } pos_dist_{q,(t_i,t_j)} < 3 \\ 0 & \text{otherwise} \end{cases}$$

NIL answers are generated as follows. If either a proper noun in a question does not appear in the top passages, or the confidence value for the top answer is below a threshold 0.1, then our system generates the NIL answer. The confidence value for NIL generation is calculated by formula (4). Finally we rank all the top answers with their confidence values.

$$Confidence_value = \frac{\sum_{(t_i,t_j) \in T} w_{q,(t_i,t_j)} \times w_{d,(t_i,t_j)}}{\sum_{(t_i,t_j) \in T} w_{q,(t_i,t_j)}} \quad \text{---------- (4)}$$

3. Evaluation Results

We obtained the following evaluation results at TREC-11.

Number of wrong answers	399
Number of unsupported answers	5
Number inexact answers	10
Number right answers	86
Confidence-weighted score	0.298
Precision of recognizing no answer	0.161 (=15 / 93)
Recall of recognizing no answer	0.326 (=15 / 46)

We selected randomly 100 of 500 questions, and evaluated the performances for some well-known answer types. The results are shown in table 3. Here, an answer type includes all subtypes of the answer type. We had a promising performance in the case of an answer type $DATE_{synset}$, but we did not generate correct answers in the case of other entity types such as $HORSE_{synset}$, CAR_{synset}.

Answer type	**Percentage**	**Precision**
person	19%	21.05%
location	23%	8.69%
organization	4%	25%
unit	7%	14.28%
count	4%	0%
date	17%	58.82%
fullname	1%	0%
other entities	25%	0%
Total	100%	18%

Table 3. Performance by Answer Types

4. Conclusion

Our QA system used WordNet as ATT for open-domain question answering. Question pattern rules were employed to determine an answer type of a question. In addition, in order to map entities in relevant passages into an entity type that corresponds to a synset in WordNet, we devised type indicator patterns. For each entity, taxonomic relationships between the answer type and its entity type are checked to qualify candidate answers. Unqualified entities are passed to an entity feedback module which provides several clue words to determine the more specific type for the problematic entity. A final answer is obtained from a ranking method, which is based on a second-order vector representation for relevant passages.

In our unpublished experiments on 300 questions in TREC-8,9,10, our system showed about 10% performance improvement by using entity feedback. But on TREC-11 questions, we did not yet perform experiments. From TREC-11 results, we had recorded a bad performance for non-basic entities like CAR_{synset}, $WEAPON_{synset}$.. In future, we plan to refine the entity classification techniques to determine types of these non-basic entities effectively, and to apply the entity feedback method on such a classification method.

Acknowledgements

This work was supported by the Korea Science and Engineering Foundation (KOSEF) through the Advanced Information Technology Research Center (AITrc).

References

[1] Cardie, C., Ng, V., Pierce, D., and Buckley C., "Examining the Role of Statistical and Linguistic Knowledge Sources in a General-Knowledge Question-Answering System", In Proceedings of the 6th Applied Natural Language Processing Conference, pp.180-187, 2000

[2] Clarke, C.L.A., Cormack, G.V., and Tudhope, E.A., "Relevance Ranking for One to Three Term Queries", Information Processing and Management, 36(2):291-311, 2000

[3] Pasca, M.A., and Harabagiu, S.M., "High Performance Question / Answering", In Proceedings of the 24rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.366-374, 2001

[4] Salton, G., and Buckley, C., "Term-weighting Approaches in Automatic Text Retrieval", Information Processing and Management, 24(5):513-523, 1988

[5] SMART system: ftp://ftp.cs.cornell.edu/pub/smart/

[6] Tree Tagger: http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/