

# Korea University Question Answering System at TREC 2004

Kyoung-Soo Han, Hoojung Chung, Sang-Bum Kim,  
Young-In Song, Joo-Young Lee, and Hae-Chang Rim

Natural Language Processing Lab.

Dept. of Computer Science and Engineering, Korea University  
1, 5-ga, Anam-dong, Seongbuk-gu, Seoul 136-701, Korea  
{kshan, hjchung, sbkim, song, jylee, rim}@nlp.korea.ac.kr

## 1 Introduction

Our QA system consists of two different components. One is for the *factoid* and *list* questions, and the other is for the *other* questions. The components are processed individually, and each result is combined into our submitted run.

For the factoid questions, we have tried to find answers by proximity-based named entity search. Given a question, fine-grained named entities for candidate answers are selected, and all the extracted passages containing the named entities and question keywords are scored by a proximity-based measure. List questions are processed in a similar way to the factoid questions, but we empirically give a threshold value to obtain only top  $n$  candidate answers.

For *other* questions, relevant phrases consisting of noun phrases and verb phrases are extracted using a dependency relationship to the question target from the initially retrieved sentences. After redundant phrases are eliminated from the answer candidates, final answers are selected using several selection criteria including the term statistics from an encyclopedia.

Section 2 summarizes our system for factoid and list questions, and Section 3 for other questions. In Section 4, the TREC evaluation results are analyzed, and Section 5 concludes our work.

## 2 Factoid and List Questions

To put it briefly, our system extracts answer candidates from AQUAINT documents according to the expected answer types of a question, and uses a confidence score to rank the answer candidates. The highest ranked answer candidate is selected as an answer for the factoid question, while answer candidates with a higher score than a

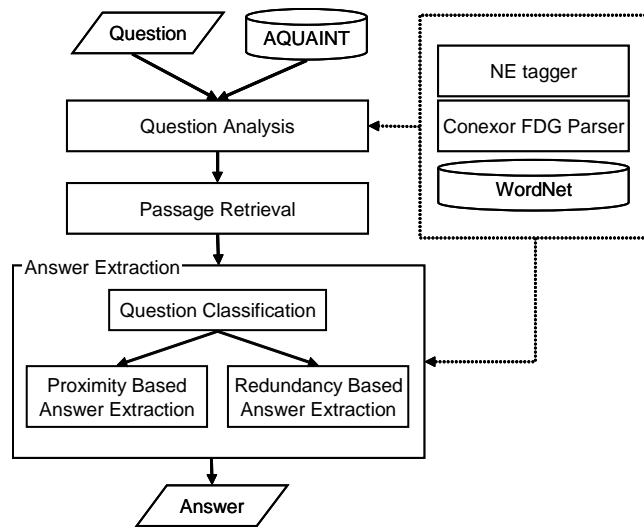


Figure 1: Overall architecture for question answering system for factoid and list questions

certain pre-defined threshold value are chosen for list questions. As shown in Figure 1, there are following three steps in answering factoid and list questions: a question analysis step, a passage retrieval step, and an answer extraction step.

## 2.1 Question Analysis

In question analysis phase, we first perform syntactic parsing using Conexor FDG parser[1]. Then we categorize given question into one of the predefined question classes. If the question is categorized into how-, when-, where- question, predefined named entity types for each question class is assigned to the given question. Otherwise, we further extract question focus, which is about what the question is about. Once the question focus is extracted, expected answer type is determined considering question focus. Finally, main verb is extracted from the syntactic information of the question sentence and heuristic rules. The main verb is used as a essential clue word in answer extraction phase.

For example, for the question “What American commodore demanded that Japan trade with the United States?”, we can obtain the following analysis results:

**QUESTION\_FOCUS** American commodore

**Expected Answer Type** PERSON

**Main Verb** demand

We manually constructed a n:n mapping table, which links between expected answer types and representative words for each type. In case of the above example, we

first extract question focus, and provide the head word “commodore” of the question focus. Since “commodore” is one of the hyponyms of `person` in WordNet and the word “person” is linked to `PERSON` in our mapping table, we assign the expected answer type `PERSON`. Thus, it is possible to assign several expected answer types given a question if the head word of the question focus have several hypernyms in WordNet.

When we define expected answer types, we consider the named entity classes provided by BBN `IdentiFinder`[2] which we used in our answer selection phase. However, since BBN `IdentiFinder` attaches only coarse-grained classes to named entities, we have further subcategorized the BBN `IdentiFinder` classes into finer-grained ones. The fine-grained named entity tagging was done with a named entity dictionary. The dictionary entries were automatically extracted from AQUAINT corpus by using simple patterns, and then some entries were added manually. Moreover, in order to cover some questions for the names of artworks such as books and movies, we have developed a title named entity tagger and tagged the title of artworks. The detail of this tagger is described in [3], and we listed some named entity tags we used in Table 1. Query words for document retrieval are also extracted in this phase by removing stopwords from the question.

## 2.2 Passage Retrieval

The relevant passages for a given question are retrieved by a document retrieval system and passage selection rules. The document retrieval system which uses Okapi ranking function retrieves AQUAINT documents relevant to the given question. Among the retrieved documents, each sentence that contains

- the expected answer type of the given question, and
- one or more question keywords, and
- any proper noun in the given question

is selected as a passage.

In addition, we also extract the passages including anaphora for the proper noun in the question. If a sentence has an appropriate anaphora for the proper noun in the question, and also contains the expected answer type and one or more keywords in question, then we check the preceding sentence. If the preceding sentence has only the proper noun in the question, we decide to extract both sentences as a passage since it is most probably that the anaphora is referring the proper noun in the preceding sentence.

## 2.3 Answer Extraction

Every named entity which matches to the expected answer type is selected as an answer candidate among the retrieved passages, which is NE tagged and dependency-parsed. Also every noun in the passages is considered as an answer candidate if the noun is a hyponym of the expected answer type in WordNet.

Category	Sub-category	Examples
Location	Country	<i>Korea, Italy, Mauritius, Mexico, ...</i>
	Continent	<i>Africa, Asia, ...</i>
	City	<i>San Francisco, Jakarta, ...</i>
	Planet	<i>Earth, Jupiter, ...</i>
	Mountain	<i>Pikes Peak, Mt. Everest, ...</i>
	Sea	<i>Red Sea, Pacific, ...</i>
Person		<i>Louis Armstrong, George Bush, ...</i>
Organization	Event	<i>Vietnam war, World War I, ...</i>
	Games	<i>Super Bowl, U.S. Open, World Cup, ...</i>
	Univ.	<i>University of Mississippi, ...</i>
	Company	<i>Coca-Cola Co., Dell, ...</i>
	Sports Team	<i>Baltimore Orioles, Seattle Mariners, ...</i>
Date		<i>July 14, ...</i>
Month		<i>April, June, ...</i>
Day		<i>Thursday, Wednesday, ...</i>
Time		<i>1:05 p.m., 7 p.m., ...</i>
Money		<i>\$35, \$5.4 billion, ...</i>
Percent		<i>10 percent, 16.5%</i>
Artworks		<i>City of Angels, Notes of a Native Son, ...</i>
Number		<i>one, two, 323, 32.35, ...</i>
	Length	<i>1,893 yards, 382 kilometers, ...</i>
	Height	<i>37-foot-high</i>
	Weight	<i>1 ton, 52kg, 0.5 gram, ...</i>
	Count	<i>3 times, 7times, ...</i>
	Speed	<i>17kph, 100Kmh,...</i>
	Temperature	<i>24 degree, 53f</i>
	Volume	<i>500cc, 1.5-liter</i>
Others		<i>Pentium III Xeon, PC, ...</i>

Table 1: Part of the named entity tag set used in our system

Based on the observation that some types of questions which contain only the question focus and target words<sup>1</sup> can be easily answered only with the co-occurrence frequency between an answer candidate and named entities in the question, we ranked answer candidates for this type of question according to their frequencies in the retrieved passages.

Otherwise, we use a confidence score for answer candidates to rank them. Our confidence score is calculated by linearly combining two scores: *surface distance score* and *syntactic similarity score*, which are described below subsections.

<sup>1</sup>For example, *Which continent is India in?* or *Who is a mayor of San Francisco?*

### 2.3.1 Surface Distance Score

We give surface distance score to each retrieved passage-answer candidate pair. The surface distance score between question  $Q$  and passage  $P$  containing answer candidate  $A$ , is calculated as follows:

$$Score(Q, P, A) = \sum_{q_i \in Q} \frac{Impact(q_i, Q) \cdot Weight(q_i t)}{Distance(q_i, P, A)} \quad (1)$$

where

$$Impact(q_i, Q) = \text{an impact factor of } q_i \text{ in } Q \quad (2)$$

$$Distance(q_i, P, A) = \min_{t_j \in rel(q_i)} \text{word distance between } t_j \text{ and candidate answer } A \quad (3)$$

An impact factor is about how important the word  $q_i$  is in question  $Q$ . We empirically set the impact factor, for instance:

- If  $q_i$  is the main verb in question  $Q$ , we set 1.5.
- If  $q_i$  has capital initial letter or  $q_i$  is superlative form, we set 1.2

In equation (3),  $rel(q_i)$  is a set of related words of  $q_i$ . This is another resource we used in answer selection. In our system, a related word of word  $t$  means a morphological derivational form of  $t$  or a special type of derivations such as (Korea, Korean) or (Europe, European).  $q_i t$  is a selected word in  $Distance$  measure, and  $Weight(q_i t)$  is 1 if  $q_i t$  is  $q_i$  itself, or 0.3 otherwise. If there is no related word of  $q_i$  in passage  $P$ ,  $Distance(q_i, P, A)$  is defined as an infinite number.

### 2.3.2 Syntactic Similarity Score

Our syntactic similarity score measures the possibility that each retrieved passage contains the proper answer for given question by comparing syntactic dependency pairs in question and passage. Among several sentences in each passage, we check only one sentence having expected answer type, that is, candidate answer.

A syntactic similarity between question  $Q$  and sentence  $S$   $SynSim(Q, S)$  is calculated as follows:

$$SynSim(Q, S) = \frac{\sum_{q_i \rightarrow q_h \in Q, w_j \rightarrow w_h \in S} weight_{dep}(q_i \rightarrow q_h, w_j \rightarrow w_h)}{|Q|}$$

where term  $q_h$  is the head of term  $q_i$  in the question, term  $w_h$  is the head of term  $w_i$  in the sentence  $S$ , and  $weight_{dep}$  means the similarity between each dependency unit in the sentence  $S$  and the question  $Q$ .

$weight_{dep}$  values are determined by checking the following heuristic rules step by step:

- (1) If  $(q_i \rightarrow q_h)$  and  $(w_j \rightarrow w_h)$  are same,  $weight_{dep}$  value for the two dependencies is 1.0.

- (2) If  $q_i = w_j$ ,  $q_h \neq w_h$ , but  $w_h$  has an indirect relation with  $q_h$  via  $n$  intermediate words,  $weight_{dep}$  of the dependencies is  $1.0 - n * 0.06$
- (3) If  $q_i$  is a synonym of  $w_j$ , or  $q_h$  is a synonym of  $w_h$ ,  $weight_{dep}$  value between the dependencies is 0.7. We have used WordNet to check whether two words have a synonym relation.
- (4) If  $q_i = w_h$  and  $q_h = w_j$ , that is, only the direction of dependency is reversed,  $weight_{dep}$  value between the dependencies is 0.6
- (5) If  $q_i = w_j$ , and it is the name of location or date,  $weight_{dep}$  value between the dependencies is 0.5

With these two score measures, we find morphologically and syntactically similar passages to a question, and the answer candidate from the passages get higher confidence scores. The highest ranked answer candidate is selected as an answer for the factoid question, while answer candidates which have higher scores than an empirically determined threshold value are chosen for the list questions.

### 3 Other Questions

Figure 2 shows the overview of our definition question answering system developed for the *other* questions. Our system extracts answer candidates consisting of noun and verb phrases related to the question target, ranks the candidates using several evidences, and returns  $n$  top-ranked candidates whose scores are higher than a pre-defined threshold.

#### 3.1 Question Analysis

Contrary to the factoid and list questions, *other* questions do not have an expected answer type such as a location or a person name. In the question analysis phase for the definition questions, the head word of the target is extracted and the target type is identified. In order to identify the head word, we syntactically parsed the question target using the Conexor FDG parser. This head word is used for a query for sentence retrieval. We have performed named entity tagging to the question target, and classified the target using the named entity of the target head word into one of three target types: person, organization, and other.

#### 3.2 Passage Retrieval

In order to answer the question, we have to retrieve relevant information to the question target in the passage retrieval phase. Because the target tends to be used with different expression in documents from that in the question, a lot of relevant information could not be retrieved or lots of irrelevant information would be retrieved by one phase passage retrieval. For example, for a target *Bill Clinton*, it would be expressed in a text by *Clinton*, *president Clinton*, *he*, etc. A strict phrase query *bill\_clinton* would suffer

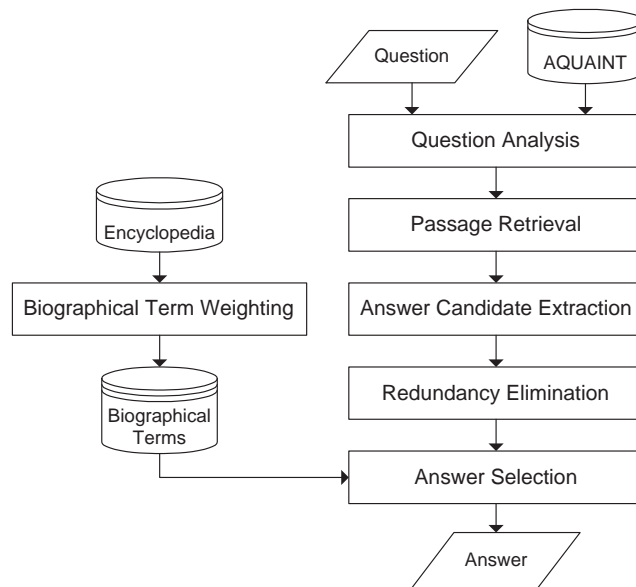


Figure 2: Overall architecture for definition question answering system

from low recall because of differently-represented phrases in documents, while a relaxed query *clinton* would be overloaded by a plenty of irrelevant information such as *George Clinton*.

Therefore, our passage retrieval engine for the *other* questions consists of two phases: document retrieval phase and sentence retrieval phase. Our intention is that we firstly retrieve only relevant documents to the target by relatively strict query, and then extract relevant sentences by using more relaxed query. The query for the document retrieval consists of the words and phrases of the target filtered with a stopword list. The phrase is used for a sequence of words with an initial capital letter, and the word itself is used otherwise. For example, for a target *Bill Clinton*, a phrase *bill\_clinton* is used as the query, and for a target *Berkman Center for Internet and Society*, the query would include a phrase *berkman\_center* and two words *internet* and *society*.

For the sentence retrieval, only the head word of the target is used as the query. In other words, sentences containing the head word are extracted from the retrieved documents. Sentences in which the head word is represented as an anaphora are also extracted by using simple rules. We observed that an anaphora refers to the target if the anaphora is used as a subject and the target is also used as a subject in the previous sentence. As shown in the following example, *he* in the sentence (b) refers to *Clinton* in the sentence (a), so both sentences are extracted.

- (a) Former president **Bill Clinton** was born in Hope, Arkansas.
- (b) **He** was named William Jefferson Blythe IV after his father, William Jefferson Blythe III.

### 3.3 Answer Candidate Extraction

For generating a more fine-grained answer, phrases are extracted from the retrieved sentences by using the syntactic dependency relations. In order to identify the syntactic relations, we also syntactically parsed the retrieved sentences using the Conexor FDG parser. Following syntactic patterns are used for extracting the answer candidates from the sentences.

**Noun phrases modifying the question target** Noun phrases that have a direct syntactic relation to the question target

**Relative pronoun phrases** Verb phrases that a nominative or possessive relative pronoun modifies directly the question target

**Participle phrases** Present or past participles that modify directly without its subject the question target or the main verb directly related to the question target.

**Copulas** Noun phrases used as a complement of a verb *be*.

**General verb phrases** Verb phrases whose head word is not stop verbs.

The stop verbs mean the functional verbs, which is not informative one such as *be*, *say*, *talk*, and *tell*.

In the above example, the following phrases would be extracted by applying the syntactic patterns:

- (1) Former president
- (2) born in Hope, Arkansas
- (3) named William Jefferson Blythe IV after his father, William Jefferson Blythe III

The syntactic information induced from the syntactic parser has many errors or there are sentences from which the information is not obtained. In order to alleviate the problem, we complement the syntactic information with POS information.

- If any word between the first word and the last of a phrase in the sentence is not extracted, it is also extracted to the phrase.
- If the last word of extracted phrase is labeled with noun-dependent POSs such as adjective, determiner and preposition, the immediate noun phrase is put together into the extracted phrase.
- If the extracted phrase is incomplete one, that is, ended with one of the POSs such as conjunction or relative pronoun, the last word is removed from the extracted phrase.

Because all extracted phrases are not useful for answer candidates, it is necessary to check the answer validity. The phrases which contain more than two content words and at least one noun or number is considered to be valid.



### 3.4 Redundancy Elimination

We also eliminated redundant information among the candidates, the extracted noun and verb phrases, based on word overlap and the semantic class of the main head word in WordNet. If two candidates share 70% or more words, the lowerly-ranked candidate is eliminated. If the word overlap does not amount to 30%, the two candidates are determined to be nonredundant. Otherwise, the semantic class of the main head word is checked. If each main head word of the two candidates is contained in a synset in WordNet, the two candidates are determined to be redundant, and the lowerly-ranked one is eliminated.

### 3.5 Answer Selection

Our system used several evidences to rank answer candidates: head word redundancy, term statistics in the relevant passages, and the biographical term weight. We have selected final answers up to 24 candidates. The highly-ranked answer candidates having a higher score than pre-defined threshold were included into the final answer.

#### 3.5.1 Head Word Redundancy

The important facts or events are usually mentioned repeatedly, and the head word is the core of each answer candidate. Therefore, we consider the redundancy of head word of answer candidate  $a$  by using following formula.

$$rdd(a) = \begin{cases} \frac{r}{|C_n|} & \text{if } a \text{ is noun phrase} \\ \frac{r}{|C_v|} & \text{if } a \text{ is verb phrase} \end{cases}$$

where  $r$  represents the number of the retrieved sentences in which the head word of the answer candidate  $a$  is used as a head word,  $|C_n|$  is the total number of answer candidates as a noun phrase, and  $|C_v|$  is the total number of answer candidates as a verb phrase. The eliminated candidates in the redundancy elimination phase also contribute to the calculation of the values.

#### 3.5.2 Local Term Statistics

In addition to the head word, the frequent words in the retrieved passages are important. The  $loc(a)$  presents a local weight based on the term statistics in the retrieved sentences, and is calculated as follows:

$$loc(a) = \frac{\sum_{t_i \in a} weight_{loc}(t_i)}{|a|} = \frac{\sum_{t_i \in a} \frac{sf_i}{maxsf}}{|a|}$$

where  $sf_i$  is the number of sentences in which the term  $t_i$  is occurred,  $maxsf$  is the maximum value of  $sf$  among all terms, and  $|a|$  is the number of all content words in the answer candidate  $a$ .

### 3.5.3 Biographical Term Weight

The biographical term weight is calculated only for the person target. The documents in an encyclopedia describe whole life of a person including personal identity and events. In other words, the encyclopedia can be a good training data for learning about the biographical definition.

The probability that a term occurs in the encyclopedia can be used for the term importance measure. The idea is that the more frequent word in the encyclopedia will be more important in the biographical definition.

$$weight_p(t) = p_{ency}(t) = \frac{tf}{T}$$

where  $tf$  is the term frequency in the encyclopedia, and  $T$  is the total occurrences of all terms.

In addition to the encyclopedia, we also utilized a general text. The term probability ratio assigns the high weight to the term occurring much more frequently in the encyclopedia than in the general text by using the following formula:

$$weight_{pratio}(t) = \frac{p_{ency}(t)}{p_{gen}(t)}$$

It is similar to the TextMap[4].

The biographical term weight is calculated using the above weights as follows:

$$bio_1(a) = \begin{cases} \frac{\sum_{t_i \in a} \log_2(weight_p(t_i) \times weight_{pratio}(t_i) + 1)}{|a|} & \text{if } a \text{ is noun phrase} \\ \frac{\sum_{t_i \in a} \log_{10}(weight_p(t_i) + 1)}{|a|} & \text{if } a \text{ is verb phrase} \end{cases}$$

$$bio_2(a) = \frac{\sum_{t_i \in a} weight_p(t_i)}{|a|}$$

Refer to the [5] for more detail comparison between these weights.

Our system KUQA1 and KUQA2 use the weights  $bio_1(a)$  and  $bio_2(a)$  respectively. KUQA3 does not use this biographical term weight.

### 3.5.4 Combination

The evidence mentioned so far is combined with linear interpolation.

$$score(a) = \alpha \times rdd(a) + \beta \times loc(a) + \gamma \times bio(a)$$

where  $\alpha, \beta, \gamma$  are tuning parameters satisfying  $\alpha + \beta + \gamma = 1$ , and are determined empirically.

The answer candidates whose score is higher than a threshold are returned as final answers. In order to eliminate the nuggets redundant to the factoid and list questions, we calculate a redundancy by content word overlap. If an answer candidate overlaps with previous questions (e.g. factoid and list questions) more than 65% or with previous answers (e.g. answers for the factoid and list questions) more than 60%, then the answer candidates are not selected as final answers.

## 4 Evaluation

We have submitted three runs : KUQA1, KUQA2, and KUQA3. For the factoid questions, KUQA1 is the result by using dependence information, while KUQA2 and KUQA3 are not.

For *other* questions, we used a public online encyclopedia, the Columbia Encyclopedia<sup>2</sup>. A part of AQUAINT corpus is used for the general text. It consists of 5,268 APW articles in July 2000, 12,843 NYT ones in March 1999, and 9,727 XIE ones in December 1998.

As shown in Table 2, we can conclude that dependence information is very useful to find an answer. Since the performances for the list questions are superior to the median performance, we can also claim that our proximity-based ranking strategy is quite effective.

Table 2: Summary of the performances

Run	factoid	list	other	final
KUQA1	0.222	0.159	0.246	0.212
KUQA2	0.187	0.157	0.229	0.190
KUQA3	0.187	0.157	0.247	0.195
Median	0.170	0.094	0.184	

For *other* questions, KUQA1 and KUQA2 used the biographical term weight,  $bio_1(a)$  and  $bio_2(a)$  respectively, based on the encyclopedia, and KUQA3 does not use the biographical term weight. It seems that  $bio_2(a)$  is not appropriate for the biographical term weight because it cannot properly normalize the score. Unexpectedly, the results show that the system KUQA1 using the encyclopedia does not outperform the system KUQA3 not using it. By analyzing the TREC evaluation results, the matching decision between KUQA1 and KUQA3 was not consistent; nuggets regarded to be matched in KUQA3 were not regarded to be matched in KUQA1. KUQA1 seems to be evaluated more strictly than KUQA3. Table 3 shows the characteristics for the two systems. In spite of the inconsistent matching decision, KUQA1 using the encyclopedia information can achieve higher precision at little cost of recall with shorter answer than KUQA3. Our system cannot answer the other questions when the parsing errors for answer-containing sentences are not successfully dealt with and any conditions for phrase extraction are not satisfied. It is necessary to relax the conditions for the more answer coverage.

Table 3: Comparison between our systems for other questions

Run	recall	precision	length
KUQA1	0.261	0.279	649.15
KUQA3	0.262	0.271	666.34

---

<sup>2</sup>The Columbia Encyclopedia. Columbia University Press, New York, 6th edition, 2004, <http://www.bartleby.com/65/>

## 5 Conclusions

We have presented an overview of our QA system for TREC 2004. Our QA system used the dependency information for all questions, and the results imply that the dependency information is useful for answer selection. Moreover, the encyclopedia was helpful to select proper answer for *other* questions.

## References

- [1] Pasi Tapanainen and Timo Jarvinen. A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64–71, 1997.
- [2] Daniel M. Bikel, Richard L. Schwartz, and Ralph M. Weischedel. An algorithm that learns what’s in a name. *Machine Learning*, 34(1-3):211–231, 1999.
- [3] Joo-Young Lee, Young-In Song, Sang-Bum Kim, Hoojung Chung, and Hae-Chang Rim. Title recognition using lexical pattern and entity dictionary. In *Proceedings of the 1st Asia Information Retrieval Symposium (AIRS)*, pages 345–348, 2004.
- [4] Abdessamad Echihabi, Ulf Hermjakob, Eduard Hovy, Daniel Marcu, Eric Melz, and Deepak Ravichandran. Multiple-engine question answering in textmap. In *Proceedings of the 12th Text Retrieval Conference (TREC-2003)*, pages 772–781, 2003.
- [5] Kyoung-Soo Han, Young-In Song, Sang-Bum Kim, and Hae-Chang Rim. Answer selection for biographical definition questions using biographical term weighting. In *Proceedings of the 1st Asia Information Retrieval Symposium (AIRS)*, pages 329–332, 2004.