TALN

19-22 AVRIL 2004 FÈS (MAROC)

XI[e] Conférence sur le Traitement Automatique des Langues Naturelles

11th Conference on Natural Language Processing

Filtering Out Bad Answers with Semantic Relations in a Web-Based Question-Answering System

Leila Kosseim and Jamileh Yousefi
CLaC Laboratory, Department of Computer Science,
1455 de Maisonneuve Blvd. West,
Montreal, Quebec, Canada H3G 1M8
{kosseim, j_yousef}@cs.concordia.ca

Mots-clefs – Keywords

Question réponse, Web comme ressource linguistique, relations sémantiques, reformulation de questions

Question Answering, Web as a linguistic resource, Semantic Relations, Question Reformulation.

Résumé – Abstract

De plus en plus de systèmes de question-réponses (QR) utilisent le Web pour trouver une réponse courte et précise à une question exprimée en langue naturelle. Dans cet article, nous présentons une méthode pour filtrer les mauvais candidats de réponses et re-ordonner les candidats dans notre module de QR en utilisant des relations sémantiques. L'idée est d'identifier la relation sémantique et l'argument principal exprimé dans la question et de trouver dans la collection de documents (ou sur le Web) d'autres indices indiquant que la relation sémantique entre l'argument de la question et le candidat de réponse existe vraiment. Les résultats avec le corpus de questions de TREC-9 et TREC-10 indique une nette amélioration de la précision de la liste des candidats.

An increasing number of Question Answering systems use the Web to find a short and precise answer to a natural language question. In this paper, we present a method to filter out noisy candidate answers and re-rank candidate answers in our Web-QA module by using semantic relations. The idea is to identify the semantic relation and the argument expressed in the question and find in the document collection (or the Web) other evidence of this semantic relation (regardless of surface form) holding between the question's argument and the candidate answer. The results with the TREC-9 and TREC-10 question sets show a net improvement of precision in the list of candidate answers retrieved.

1 Introduction

To improve the QUANTUM QA system, we have included a Web module that searches the Web for candidate answers, that are then combined with QUANTUM answers found in a small collection to confirm a possible answer (Plamondon *et al.*, 2001; Plamondon *et al.*, 2002). Similarly to many Web-based QA systems (Clarke *et al.*, 2001; de Chalendar *et al.*, 2002), our Web module uses simple question re-write rules to generate answer contexts that are restrictive enough for finding answers on the Web. Searching for such contexts and then performing simple semantics checks on the extracted answers leads to a correct answer for 25 % of a set of questions from the TREC-9 and TREC-10 conferences (Plamondon & Kosseim, 2003).

Although in conjunction with QUANTUM, the Web-component yields an interesting improvement, alone its results are very noisy. The right answer is there, but hidden along side many wrong answers. Without the core QUANTUM system, no distinction between right and wrong answers is possible.

In this paper, we present a method to filter out noisy candidate answers and re-rank candidate answers in our Web-QA module by using semantic relations. The idea is to identify the semantic relation and the argument expressed in the question and find in the document collection (or the Web) other evidence of this semantic relation (regardless of surface form) holding between the question's argument and the candidate answer.

In section 2, the components of the Web-QA are explained. In Section 3, we describe our techniques for filtering out bad answers and re-rank the results. Finally, in section 4, we evaluate our approach with the questions from the TREC-9 and TREC-10 collection.

2 The Original Web-QA Component

Similarly to many Web-based QA systems (Poibeau *et al.*, 2003), our Web-QA module uses answer formulation to drive the search for answers (Plamondon & Kosseim, 2003; Plamondon *et al.*, 2002). That is, we search the Web for an exact phrase that could be the formulation of the answer to the question. For example, given the question *Who is the prime minister of Canada?*, our goal was to produce the formulation `The prime minister of Canada is` <PERSON-NAME>. Then, by searching the Web for this exact phrase and extracting the noun phrase following it, our hope is to find the exact answer. Simple syntactic and semantic checks were then performed to ensure that the following noun phrase is indeed a PERSON-NAME.

To formulate an answer pattern from a question, we turn the latter into its declarative form using a set of hand-made formulation templates that test for the presence of specific keywords, grammatical tags and regular expressions. Figure 1 shows an example of reformulation. The formulation template is composed of 2 sets of patterns: A question pattern that defines what the question must look like, and a set of answer patterns that defines a set of possible answer formulations. The patterns take into account specific keywords (e.g. `When did`), strings of characters (e.g. `ANY-SEQUENCE-WORDS`) and part-of-speech tags (e.g. `VERB-simple`). Answer patterns are specified using the same type of features plus a specification of the semantic class of the answer (e.g. `TIME`). The semantic classes are used later, during answer extraction, to validate the nature of the candidate answers from the document.

Formulation Template	Example
`When did ANY-SEQUENCE-WORDS-1 VERB-simple ?`	*When did the Jurassic Period end?*
`ANY-SEQUENCE-WORDS-1 VERB-past TIME`	`the Jurassic Period ended TIME`
`TIME ANY-SEQUENCE-WORDS-1 VERB-past`	`TIME the Jurassic Period ended`
`TIME, ANY-SEQUENCE-WORDS-1 VERB-past`	`TIME, the Jurassic Period ended`

Figure 1: Example of a formulation template

Corpus	No questions	Nb of questions with a reformulation	Nb of questions with at least one candidate answer	Nb of questions with a correct answer in the top 5 candidates	Precision of candidate list
TREC-9	694	624 (89.9%)	63 (9.1%)	17 (2.4%)	0.270
TREC-10	499	450 (90.2%)	256 (51.3%)	153 (30.7%)	0.598

Table 1: Results of the Web-QA component alone

In total, 77 formulation templates are used. The templates are tried sequentially and all question patterns that are satisfied are activated.

Although the Web component was meant to complement our core QA system, when evaluated on its own, its results were very low. Table 1 shows the evaluation of the Web-QA component alone with the TREC-9 and TREC-10 data. Note that these figures were computed by taking the first 200 candidates retrieved from the Web. Although most questions from the data set were actually covered by the reformulation patterns and generated at least one reformulation (see column 3), only a small number of reformulations actually retrieved a correct answer in the top 5 candidates (see column 5) and the precision of the list of candidate answers is rather low (see column 6). For example, with the TREC-9 question set, only 27% (17/63) of the questions for which a list of candidates was retrieved contain a correct answer in the top 5 candidates. Our goal was then to improve these results by filtering out the *noisy* candidates and re-rank the remaining candidates better.

3 Filtering out bad answers

To filter and better rank the results of the Web-QA component, we were inspired by the approach used in the WebStorm System (Duclaye *et al.*, 2002; Duclaye *et al.*, 2003). In WebStorm, the authors used the Web as a linguistic resource to learn reformulations automatically. They start with one single prototypical argument tuple of a given semantic relation and search for potential alternative formulations of the relation, then find new potential argument tuples and iterate this process to progressively validate the candidate formulations. Their approach focuses on the use of paraphrases as a potential way to improve Question Answering systems. Similarly to their work, we extract the semantic relation expressed in the question, and create argument tuples from the noun phrase expressed in the question and the noun phrases that the Web-QA module identifies as candidates. We then run the Web-QA module again, but this time, only

these tuples are used to try to find new evidence of the original relation, expressed in various linguistic form. To do so, we use WordNet (Miller, 1995) and the Porter stemmer (Porter, 1980) to identify semantically related verbs, nouns and adjectives. Let us describe this process in details.

Initial Run We first run the Web-QA component and retrieve its top 200 candidates. For example, for *'Who killed Martin Luther King?'*, the following candidate answers are retrieved by Web-QA:

```
Bobby Kennedy
Activism Exit RW ONLINE Who
Who Really
Crawford The Real Reason They
James Earl Ray
Dream Who n
```

The candidate answers are then tagged by the GATE-NE named-entity tagger (Cunningham *et al.*, 2002), and the ones which satisfy the predicted constraints of the answer are kept. For example for the above question, only candidates containing a <PERSON-NAME>s are chosen:

```
Bobby Kennedy
James Earl Ray
```

Finding the semantic relation We then decompose the original question into two parts: the main semantic relation expressed (e.g. *killed*) and the argument of the relation (e.g. *Martin Luther King*).

The semantic relation is taken to be the main verb of the question that describes a relation between the question argument and the candidate answer proposed by the Web-QA module. For example, in *'Who invented television?' 'invented'* is taken to be the relation between the candidate answer and *'television'*. In essence, we are trying to represent the question in the logical form `invented(television,` ANSWER`)`. Note that our current implementation only considers relations between two arguments. Relations holding between more than two arguments, for example *'When did ARG1 give ARG2 to ARG3?'* are considered as a binary relation between ARG1 and the candidate answer. This assumption should only lead to loss of precision.

If the question only contains semantically weak verbs (e.g. to-be, or modal verbs), the verb is not considered as the semantic relation of the question. For such question, only the argument and the candidate answer are considered for validation and filtering. For example, in *'Who is the president of the US?' is* will not be considered as a semantic relation.

Filtering and Re-ranking candidates A set of argument tuples are then created from the argument of the question and the candidates found by the Web-QA component. In our case, the following tuples are created:

```
(Martin Luther King, Bobby Kennedy)
(Martin Luther King, James Earl Ray)
```

Once we have built the set of argument tuples, we search them in the document collection to

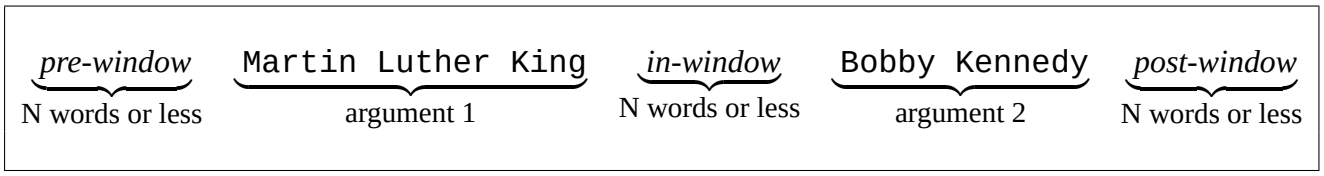

Figure 2: Example of a context window for the argument tuple (Martin Luther King, Bobby Kennedy)

identify the possible semantic relations holding between them, and make sure that the relation that relates them in the documents is equivalent to what we were originally looking for in the question.

In our experiment, we submitted all the tuples to both the TIPSTER collection and the Web to find paragraphs that contain these tuples. Then we extracted only the paragraphs where both tuple elements were at a distance of N[1] words or less. We used a context window size of N words between the tuple elements and N words on each side of them in the extracted paragraphs and then examined the words in these context windows for a possible similar semantic relation. This is shown in Figure 2.

For example, the excepts found in the TIPSTER document collection for the tuple (Martin Luther King, James Earl Ray) are:

```
...strike of black garbage workers.  James Earl Ray killed
Martin Luther King and pleaded guilty.  Memorials ...

...A small-time thief named James Earl Ray shot Martin Luther King
from the bathroom of the flophouse...

...there is a Likelihood that James Earl Ray assassinated Dr.
Martin Luther King, JR., as a result of...
```

Finally, we analyze the words in the context windows to identify if at least one is semantically equivalent to the original semantic relation expressed in the question. To verify this, we first check if any verb found in any context window is a synonym, a hypernym or a hyponym of the original verb in the question. For such a task we have used a part-of-speech tagger and WordNet.

If no verb has an equivalent semantic relation, we then backup to analyzing other parts of speech. We try to validate nouns and adjectives. We use the Porter stemmer (Porter, 1980) to find the stem of the words and we check if it has the same stem as the original verb or one of its synonyms. For example, if the phrase *the assassination of* appears in a context window, we check if the original verb `kill` in the question or one of its synonyms share the same stem with *assassination*. The stem *assassin* is indeed the same as the stem of the synonym *assassinate*.

For example, the excepts found in the Web for the tuple (Martin Luther King, James Earl Ray), with other parts of speech carrying the semantic relation, are:

[1] In our experiment, N was set to 5.

```
...8, 1968 In History, Event James Earl Ray, alleged assassin of
Martin Luther King Jr, is captured at a...

...Killing the Dream:  James Earl Ray and the Assassination of
Martin Luther King, Jr.  by Authors:  Gerald Posner...
```

Any tuple that cannot be found to have a similar semantic relation in the question and in the documents is thrown out. For example, in the following passages found on the Web for the tuple `(Martin Luther King, Bobby Kennedy)`, none contain a word that is semantically equivalent to the relation expressed in the question:

```
...shame men like Derwin Brown, Martin Luther King, Jack Kennedy,
Bobby Kennedy, and many others are taken...

...Rosa Parks, and John and Bobby Kennedy.  Dr.  Martin Luther King,
Jr.  was the leader of...
```

The remaining candidates are then re-ranked according to the number of passages in the collection containing the same relation. For example, when we submitted the tuple `(Martin Luther King, James Earl Ray)` to the TIPSTER collection, we found 110 passages containing the elements of the tuple. Among these passages, only 24 contained the tuples and the relation `kill` within 5 words of each other. We therefore gave a rank of (24/110) to the candidate `James Earl Ray`. By applying this procedure to all the argument tuples, the five best ranked candidates can be easily found and selected.

4 Evaluation

We evaluated our approach with the questions from the TREC-9 and TREC-10 collection and compared the answers found this way with the original candidates. Table 2 shows the results of this evaluation. Although the number of questions with at least one candidate answer (column 4) is inferior in the new system and the number of correct candidates (or actual answers) is similar, the precision of the candidate list is much higher. This means that, although we provide less candidates, they are more likely to constitute correct answers than before.

If we look closer at the list of candidates, very few correct answers were (wrongly) discarded by the semantic filtering. With the TREC-9 questions, 5% of the good answers were removed from the list of candidates, while no good answer was lost for TREC-10.

With the TREC-9 questions, 25% of the correct answers were ranked better, by moving up the list by an average of 2.2 positions. However, 5% of the correct answers were demoted to a lower rank (on average 6 positions down). Overall, the re-ranking method improved the mean reciprocal rank (MRR) by 40% for TREC-9.

With the TREC-10 questions, 31% of the correct answers were ranked better by an average of 4.48 positions, but 9% ranked worse by an average of 3.5 positions. Overall, the re-ranking method improved the MRR by 12% for TREC-10.

Corpus	System	Nb of questions	Nb of questions with at least one candidate answer	Nb of questions with a correct answer in the top 5 candidates	Precision of candidate list
TREC-9	Original system	694	63 (9.1%)	17 (2.4%)	0.270
TREC-9	New system	694	28 (4.0%)	20 (2.9%)	0.714
TREC-10	Original system	499	256 (51.3%)	153 (30.7%)	0.597
TREC-10	New system	499	189 (37.8%)	152 (30.5%)	0.804

Table 2: Results with the original version of the Web-QA component and the current version

5 Discussion and Future Work

The method described here improves the accuracy of our Web-QA module by re-ranking the candidates and by discarding those that do not contain the correct semantic relation.

As opposed to several other approaches that reinforce their candidate answers by looking on the Web; our approach is less strict as it looks for reinforcement of the semantic relation between the arguments, rather than looking only for lexically similar evidence. In this respect, our approach is much more tolerant and allows us to find more evidence. On the other hand, as we look for evidence in a window of N words, rather that a strict string match, we are more sensitive to mistakes and wrong interpretations. Indeed, we are only interested in finding a word that carries a similar sense without doing a full semantic parse of the sentence. Negations and other modal words may completely change the sense of the sentence, and we will not catch it. When looking in a very large corpus such as the Web, this may lead to more noise than a strict lexical string match approach. However, if we perform the QA task on a much smaller corpus, such as in closed-domain QA, looking for semantic equivalences may be more fruitful.

As mentioned in section 3, the current implementation only looks at semantic relations holding between pairs of arguments. However, it can easily be extend to consider variable-size relations. However, as more constraints are taken into account, the precision of the candidate list is expected to increase, but recall is expected to decrease. An careful evaluation would be necessary to ensure that the approach does not introduce too many constraints and consequently filters out too many candidates.

As table 1 shows, another important problem in our current Web-QA system is that a large number of questions that are reformulated retrieve no candidate answer (compare column 3 with column 4). Our next goal is now to look at generating better reformulations so that the system can retrieve candidates for more questions.

Acknowledgments

This project was financially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and Bell University Laboratories (BUL). The authors would like to thank the anonymous referees for their constructive comments.

References

CLARKE C., CORMACK G., LYNAM T., LI C. & MCLEARN G. (2001). Web Reinforced Question Answering (MultiText Experiments for TREC 2001). In *Proceedings of The Tenth Text Retrieval Conference (TREC-X)*, p. 673–679, Gaithersburg, Maryland.

CUNNINGHAM H., MAYNARD D., BONTCHEVA K. & TABLAN V. (2002). GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, p. 168–175, Philadelphia.

DE CHALENDAR G., DALMAS T., F. E.-G., FERRET O., GRAU B., HURAULT-PLANTET M., ILLOUZ G., MONCEAUX L., ROBBA I. & VILNAT A. (2002). The Question Answering System QALC at LIMSI: Experiments in Using Web and WordNet. In *Proceedings of The Eleventh Text Retrieval Conference (TREC-11)*, p. 457–465, Gaithersburg, Maryland.

DUCLAYE F., YVON F. & COLLIN O. (2002). Using the Web as a Linguistic Resource for Learning Reformulations Automatically. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, p. 390–396, Las Palmas, Spain.

DUCLAYE F., YVON F. & COLLIN O. (2003). Learning paraphrases to improve a question-answering system. In *Proceedings of the Natural Language Processing for Question Answering Workshop at EACL (EACL'03)*, Budapest, Hungary.

MILLER G. (1995). WordNet: a Lexical Database for English. *Communications of the ACM*, **38**(1), 39–41.

PLAMONDON L. & KOSSEIM L. (2003). Le Web et la question-réponse: transformer une question en réponse. In *Actes des Journées Francophones de la Toile (JFT-2003)*, p. 225–234, Tours, France.

PLAMONDON L., LAPALME G. & KOSSEIM L. (2001). The QUANTUM Question Answering System. In *Proceedings of The Tenth Text Retrieval Conference (TREC-10)*, p. 157–165, Gaithersburg, Maryland.

PLAMONDON L., LAPALME G. & KOSSEIM L. (2002). The QUANTUM Question Answering System at TREC-11. In *Proceedings of The Eleventh Text Retrieval Conference (TREC-11)*, p. 157–165, Gaithersburg, Maryland.

POIBEAU T., ZWEIGENBAUM P. & NAZARENKO A. (2003). Traitement automatique des langues pour les systèmes de question/réponse. In *Journée RIP-WEB*. http://www.limsi.fr/Individu/monceaux/RIP-Web/qa.pdf.

PORTER M. (1980). An algorithm for suffix stripping. *Program*, **14**(3), 130–137.