A prototype Question Answering system using syntactic and semantic information for answer retrieval

Enrique Alfonseca, Marco De Boni, José-Luis Jara-Valencia, Suresh Manandhar

Department of Computer Science
The University of York
Heslington
YORK
YO10 5DD
United Kingdom

Introduction

This was our first entry at TREC and the system we presented was, due to time constraints, an incomplete prototype. Our main aims were to verify the usefulness of syntactic analysis for QA and to experiment with different semantic distance metrics in view of a more complete and fully integrated future system. To this end we made use of a part-of-speech tagger and NP chunker in conjunction with entity recognition and semantic distance metrics. We also envisaged experimenting with a shallow best first parser but time factors meant integration with the rest of the system was not achieved. Unfortunately due to time constraints no testing and no parameter tuning was carried out prior TREC. This in turn meant that a number of small bugs negatively influenced our results. Moreover it was not possible to carry out experiments in parameter tuning, meaning our system did not achieve optimal performance. Nevertheless we obtained reasonable results, the best score being 18.1% of the questions correct (with lenient judgements).

Question-Answering algorithm

The YorkQA question answering system takes inspiration from the generic question answering algorithm presented in [Simmons 1973]. Although unacknowledged by more recent research its general structure is very similar to the basic algorithms used in the Question Answering systems built for previous TREC conferences

The algorithm proceeds as follows:

I) Accumulate a database of semantic structures representing sentence meanings.

II) Select a set of structures that appears relevant to the question. Relevance is measured by the number of lexical concepts in common between the proposed answer and the question. This is done by ordering the candidates according to the number of Token values they have in common with the questions.

III) Match the question structure against each candidate. This is done by:

- Determining if the head verb of the question matches the head verb of the candidate. If there is no direct match, a paraphrase rule is applied to see if the question structure can be transformed into the structure of the answer. Paraphrase rules are stored as part of the lexicon and an examination of the lexical structures of two words will be able to determine if there is a rule (path) connecting the two. If there is not, the set of words that the first transforms into is recursively examined to see if can be transformed into the second word. If this fails, the transformation rules are recursively applied to the second word to see if a match can be found. This procedure continues until either a match is found or an arbitrarily set depth is reached.
- Applying the same procedure to the other words in the question and the candidate answer question in order to transform the question structure into the form of the candidate answer.
- Examining quantifiers and modalities to see if quantificational, tense and negation relationships are matched.
- Examining the question's semantic structure to determine if the question word type (the wh-word) is present and satisfied in the answer

Our algorithm proceeded as follows:

1. Index the documents using a standard Information Retrieval engine
2. Read in the next question, and repeat the following until there are no more questions:
3. Analyse each question to determine the question category, i.e. the type of entity the answer should contain
4. Decide the query to send to the IR engine and send the query
5. Pass the retrieved documents to the sentence splitter, tagger, chunker and named entity recogniser
6. Analyse the sentences to determine if they contain the correct entity type.
7. Rank the sentences containing the correct entity type according to their semantic distance from the question.
8. Use a parser to determine the answer by finding a match between the question and the ranked sentences
9. If matching through the parser fails look for the best entity in the top ranked sentences or, if an entity of the appropriate type is not found, an appropriate 50 byte window.

Step 1 of our algorithm corresponds to point I in Simmons' algorithm, 2-7 correspond to II and 8-9 correspond to III. Due to time constraints we were unable to implement 8.

What follows is a more detailed description of the main parts of our system.

Question Type Identification

The question analyser used pattern matching based on wh-words and simple part-of-speech information combined with the use of semantic information provided by WordNet [Miller 1995] to determine question types. Unlike other question analysers constructed for previous TREC QA tracks (e.g. [Harabagiu 2001]) our question type analyser only recognised a small number of different types. Question types were limited to time, place, currency, organisation, length, quantity, reason, place, constitution, person, length, mode, recommendation, truth and a fallback category of thing. A number of these, for example recommendation ("Should I do X?"), reason ("Why did X occur") and truth ("Is X Y?") revealed themselves not to be needed for the track as no questions of that type were present. An initial evaluation estimated an accuracy of 83% for the question recogniser.

Information Retrieval

Steps 1 and 3 of our algorithm were carried out by a) using the SMART Information Retrieval system (for an introduction to SMART, see [Paijmans 1999]) to index the documents and retrieve a set of at most 50 documents using the question as the query; and b) taking the first 50 documents supplied by the Prise Information Retrieval engine, as provided by the TREC organisers. Unfortunately due to time constraints we were unable to tune the SMART IR system for the task at hand and in fact the documents retrieved by the SMART engine were much worse (very low in both precision and recall) than the documents provided by NIST.

The YorkQA system then transformed the output of the retrieval engine into appropriate XML documents and used a number of tools for linguistically processing them, in particular a tokeniser, sentence splitter, part-of-speech tagger, morphological analyser, entity recogniser, QP- and NP-chunker.

Tagging and chunking

The texts were processed using the TnT part-of-speech tagger [Brants, 2000] and a Noun Phrase chunk parser based on Transformation Lists [Ramshaw and Marcus, 1995] to find non-recursive noun phrases. To improve the accuracy of the chunker, we trained it on a copy of the Wall Street Journal corpus that was manually corrected by a human annotator, so as to improve the quality of the training data. This increased the initial accuracy more than 3% to 95%.

The sentence splitter was based on [Mikheev, 1999]. It is divided in two steps of processing dot-ending words. Firstly, the system decides whether they are or not abbreviations. Non-abbreviations ended with dots all indicate sentential endings; and abbreviations followed by a non-capitalised word are never sentence ends. The second step addresses the difficult case, when an abbreviation is followed by a capitalised word, and several heuristics are used to decide whether a sentence ends there or not.

Named Entity Recognition

This was an important step in the processing of the text as the YorkQA system initially tried to find sentences containing an appropriate entity that might answer a determined question. Nevertheless this module was, for this version, the weakest link in the processing pipeline and should, and will, be subject to serious improvement in the future. At present, this tool aims to recognise six types of entities:

- *Currency expressions*, such as "$10,000", "2.5 million dollars", etc. The evaluation shows a very high accuracy for this sub-module (4/4), but all the expressions found were in dollars and it is not clear that this performance could be obtained for other types of currencies. Its precision however should remain very high in any stricter evaluation.
- *Locations*, such as "Chicago", "Thames", "Mount Kilimanjaro", etc. This sub-module is programmed to recognise locations by context words, so that it recognises "New York City" but not "New York". This limitation is clearly reflected in its performance as this module missed all location expressions (0/32) in the sentences selected for the manual evaluation.
- *Organisations*, such as "Climbax Corp.", "Nobel Prize Committee", "The National Library of Medicine", etc. The approach used by this sub-module is to look for a clear organisational word at the end of a sequence of capitalised words. Therefore, it recognises "National Cancer Institute" but not "Massachusetts Institute of Technology". Consequently, we determined that only 35% (8/23) of the organisations mentioned in the evaluated text were correctly tagged and that most of the error were due to poor recall.
- *People names*, such as "Reagan", "Marilyn Monroe", "G. Garcia Marquez", etc. Again precision is preferred to recall in this sub-module. Therefore, people's names are marked when they are surrounded by clear context words such as personal titles ("Mr.", "Dr.", etc.) and common pre-name positions ("President ...", "Sen. ...", etc.) or both the forename(s) and the surname(s) are found in a gazetteer of common names in several languages. The evaluation is consistent with this bias as most of the 69% of incorrect answer (13/42) from this module are consequence of poor recall.
- *Quantity expressions*, such as "twelve", "11/2 billion", etc. Because of the relative regularity in this kind of expressions, this sub-module gets a fairly good accuracy (60/73) and most errors are misinterpretations of the words "a" and "one".
- *Time expressions*, such as "June, 28 1993", "Today", "late 2000", etc. The accuracy of this sub-tool is around 50%, mainly because it misses many relative time expressions (such as "the day after the great storm") which are difficult to capture by regular expressions only.

Clearly these expressions help the system to answer questions about people, organisations, money, etc. but we are very aware that much more must be done on this. For example, the system would get better performance if a broader range of entities were recognised, in particular speeds, weights, lengths, duration expression, etc. will certainly increase the focus of the search which at present can only rely on quantity entities. This and other improvements are planned in the future and we hope to have in hand a much better entity recogniser for the next version of YorkQA.

Measuring Semantic distance

The central part of our system was the semantic distance measuring module, which analysed the tagged and chunked sentences in the documents selected by the Information Retrieval engine, comparing them with the question in order to find the sentence which was most similar.

To calculate similarity was necessary to calculate the semantic (or conceptual) distance between a sentence and a question. A number of algorithms have been presented to measure semantic (or conceptual) distance (see for example [Miller and Teibel 1991, Rada et al. 1989]). WordNet [Miller 1995] has been shown to be useful in this respect, as it explicitly defines semantic relationships between words (see, for example, [Mihalcea and Moldovan 1999] and [Harabagiu et al. 1990]). Our system, however, differs from previous approaches in that it does not limit itself to considering the WordNet relationships of synomymity and hyperonymy, but makes use of all semantic relationships available in WordNet (is_a, satellite, similar, pertains, meronym, entails, etc.) as well as information provided by the noun phrase chunker.

Answer identification

The question types were used to identify a candidate answer within the retrieved sentences in the case of questions of type quantity, person, place, time and currency; in the case of questions of type reason, recommendation, mode and difference, appropriate keywords were searched for (e.g. a part of sentence following the word "because" is probably a reason); finally, if the question analyser failed to recognise any question type (i.e. the answer type was the catch-all category "thing"), the system looked for a portion of text 50 bytes long within a sentence which was closest to the words contained in the question, but contained the lowest number of question words.

Results, Conclusions and further work

Given that our system was a simple prototype which had not been tested, nor tuned, the results we got were encouraging, the best score being 18.1% of the questions correct (with lenient judgements), proving that the use of syntactic and semantic information can be fruitfully used for the QA task.

The current system will have to be fully tested and debugged and a full evaluation will have to be carried out on each separate module to evaluate performance and identify the source of errors.

Future work will include: generic question type identification (needed since handing coding for each type is cumbersome and time consuming); an improved named-entity recogniser; improved semantic distance metrics; a parser to transform sentences into a simple logical form which can then be manipulated in order to find an answer.

Bibliography

Brants, T.,"TnT - A Statistical Part-of-Speech Tagger", User manual, 2000

Harabagiu, S. A., Miller, A. G., Moldovan, D. I., "WordNet2 - a morphologically and semantically enhanced resource", In Proceedings of SIGLEX-99, University of Maryland, 1999.

Harabagiu, S.,et al., FALCON - Boosting Knowledge for Answer Engines. In Proceedings of TREC-9, NIST, 2001.

Mikheev, Andrei, "Periods, capitalized words, etc." Submitted to Computational Linguistics, 1999

Miller, G, and Teibel, D., "A proposal for lexical disambiguation", in Proceedings of DARPA Speech and natural Language Workshop, California, 1991.

Miller, G. A., "WordNet: A Lexical Database", Communications of the ACM, 38 (11), 1995.

Paijmans, Hans, "SMART Tutorial for beginners.", http://pi0959.kub.nl:2080/Paai/Onderw/Smart/hands.html, 1999

Rada, R., Mili, H., Bicknell, E. and Blettner, M., "Development and application of a metric on semantic nets", in “IEEE Transactions on Systems, Man and Cybernetics”, vol.19, n.1, 1989.

Rada Mihalcea and Dan Moldovan, A Method for Word Sense Disambiguation of Unrestricted Text, in Proceedings of ACL '99, Maryland, NY, June 1999.

Ramshaw, Lance A. and Marcus, Mitchell P. , "Text Chunking Using Transformation-Based Learning", Proceedings of the Third ACL Workshop on Very Large Corpora", 82--94, Kluwer, 1995.

Simmons, R. F., "Semantic Networks: computation and use for understanding English sentences", in Schank, R. C. and Colby, K. M., "Computer Models of Thought and Language", San Francisco, 1973.