TREC-9 Question Answering: Some general and some specific issues

Nov. 13–16, 2000

Vlado Kešelj, December 1, 2000

Department of Computer Science, University of Waterloo,
Waterloo, ON N2L 3G1, Canada,
`vkeselj@cs.uwaterloo.ca`, `http://www.cs.uwaterloo.ca/~vkeselj`

Abstract. This is an informal incomplete report on the TREC-9 Question Answering track, which includes the following topics:

- the question-answering problem (track) formulation for TREC-9,
- QA track results,
- some common terminology and system components across the QA systems (participating in TREC),
- future TREC QA directions for the next five years,
- description of the best-performing system FALCON, and
- comparing the FALCON logical form with a logical form based on attribute value matrices (i.e., feature structures).

1 Problem definition

The question-answering (QA) problem in TREC was defined in TREC-8, when the QA track is introduced. Given a natural language (NL) question (in English), the system is supposed to find a relevant document for the question, and extract a short answer string that contains an answer to the question. Two limits for answer-string lengths are used: 50 bytes and 250 bytes.

For each question, a system returns 5 ranked answer strings. The evaluation metric is reciprocal answer rank (RAR), i.e., a question receives a score equal to the reciprocal of the rank at which the first correct answer string (or response) is returned, or 0 if none of the five responses contained a correct answer.

There were 500 questions collected from the log of Microsoft Encarta, and the Excite log. It was verified that each question has an answer in the collection. Additional 193 questions are syntactic variants of 54 questions. That gives a total of 693 question. However, during evaluation it was realized that there are no answers for some questions, so the final evaluation is done on 682 questions.

It was important that the answer strings are found in relevant documents, more precisely, in the documents that support given answers.

Twenty-eight groups participated, submitting 78 runs (34 50-byte-limit runs and 44 250-byte-limit runs) [VH00].

2 Approaches

The approaches can be divided in two groups, which can be called:

- statistically-based approaches, and
- knowledge-based approaches.

The distinction is not clear, and it could be argued that all approaches are more-or-less of hybrid type. However, they can be compared according to the relative emphasis that they put on the statistical or knowledge-based methods.

The following table lists 22 groups that submitted 50-byte-limit runs, with their best results ([TRE00]).

rank	score	group	know. vs stat.
1.	0.580	SMU	K
2.	0.321	Waterloo	S
3.	0.318	USC (21 aal)	K
4.	0.315	IBM T.J. Watson	S
5.	0.284	Queens	S
6.	0.249	Syracuse	K
7.	0.231	NTT	S
8.	0.231	Imperial College	?
9.	0.230	Alicante University	K
10.	0.227	Xerox	?
11.	0.212	KAIST	S
12.	0.206	Sheffield	K
13.	0.196	Microsoft (25 aal)	K
14.	0.195	Fudan University	S
15.	0.180	Alberta	?
16.	0.179	Montreal	S
17.	0.178	LIMSI	S
18.	0.135	CL Research (10 aal)	K
19.	0.101	Seoul	S
20.	0.084	Pisa	K
21.	0.064	Conexor Oy	?
22.	0.038	MITRE	?

The last column in the table represents knowledge-based versus statistical approach classification. Some groups submitted runs with an average answer-string length significantly lower than 50, and that is denoted in parentheses, with the abbreviation 'aal'.

As mentioned above, all groups used some techniques that may be called knowledge-based, and all groups used some techniques that may be called statistical. A minimal use of knowledge-based methods includes some knowledge about the natural language, and a minimal use of statistical methods includes some scoring techniques.

Knowledge-based and statistical approaches are generally intermixed in the ranked list presented in the table. However, the best result obtained by Southern Methodist University, which represents a knowledge-based approach, is significantly higher than then next result. The results are graphically shown in Figure 1.

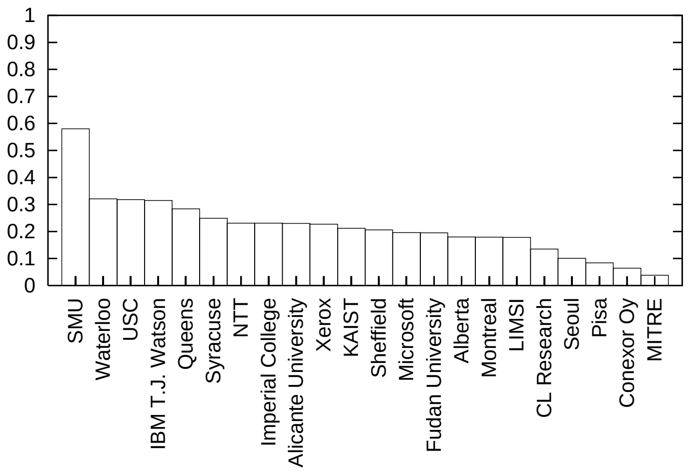


Fig. 1. QA results

The common terminology and the common components across the systems are: A *question parser* is used to parse a question, and to determine *question category*, i.e., the *answer category*. Using the question, and possibly the additional information obtained from parsing, the *passage retrieval* is performed. *Named entity recognizer* is used to annotate passages, i.e., to recognize *named entities.* Finally, a component called *answer extractor* is used to determine final list of answer strings.

3 Future

The following list of new features will appear in TREC-10 question answering:

– **list answers:** some questions will require a list as an answer, it is not yet clear how these questions will be scored;
– **no answer:** if a question cannot be answered using the given collection, then the string "no answer" is the correct answer,
– **exact answer:** instead of having 50-byte-limit and 250-byte-limit runs, there will be an exact-answer, and a 50-byte-limit run,

– **deduction-based answers:** questions that require some primitive level of deduction, possibly from multiple documents; e.g., which of the two given cities has larger population, and
– **compound answers:** e.g., questions like "When and where did *smth* happen?".

The last two features, deduction based answers and compound answers, will be separated in a separate group of questions. A participating group can choose not to tackle those questions.

The TREC-QA tentative plan for the next five years is the following:

TREC-10 (2001) is described above;
TREC-11 (2002) will pursue answers within context, follow-up questions, references to the previous answer;
TREC-12 (2003) will include text generation, explaining ambiguities, giving justification;
TREC-13 (2004) will include summary-generating questions:
1. context-based summary-generating questions,
2. stand-alone summary-generating questions, and
3. example-based summary-generating questions;

TREC-14 (2005) will include expert-level questions.

4 SMU system description ([HMP+00])

General. The system represents a knowledge-based approach. The external resources used are WordNet and the Collins's probabilistic parser. Several knowledge bases are used to improve performance. Some scoring technique is used, but it is not reported. The system techniques are described as: "surface-based NLP techniques that are boosted with pragmatic knowledge that filters out incorrect answers."

Question reformulations and caching. The system attempts to detect question reformulations, which were known to be included in the question set. Each new question is compared to previous questions. If the questions are close enough, then already found answers to the previous question are returned. The measure of question similarity is based on comparing similar words with equal POS tags. There are several loops: in the first one, 'similar' means equal, and in the later loops one of the three relaxations are allowed: same stem, WordNet synonyms, and WordNet hypernyms.

Passage retrieval. A modified SMART system that supports boolean queries is used for passage retrieval. If a number of retrieved passages is not satisfying (too low or too high), then the retrieval is repeated with a modified query. If there were to many passages, new keywords are added; and if there are too few passages, some keywords are deleted. This is the first of the three large loops in the procedure.

In order to reduce the number of loop iterations—this one and the later ones—morphological alternations are initially included in query, if the original term is very specific. 'Specificity' is measured using WordNet: it is the number of IS-A hypernym relations.

Parsing questions and passages. Collins's probabilistic parser is used to parse questions, and, apparently, passages. From a parse tree of a question, a dependency tree is constructed, in which some closed-set words are skipped. The node with 'largest connectivity' determines the search concept, i.e., the *expected answer type.*

Answer types and named entity categories. A large hierarchy of answer types is used. There is an initial taxonomy, which is extended by some WordNet sub-hierarchies. The expected answer type of a question is determined as the top type (the most specific super-type) of a set of candidates. The motivation for using WordNet is to extend named entity recognition to deal with the semantic class of any concept lexicalized in English and encoded in WordNet.

The *named entity recognizer* annotates passages with named *entity categories.* There are 27 named entity categories. The mapping between answer types and named entity categories is many-to-many.

Question and passage logical forms. The dependency trees obtained from the parser are translated to logical forms, which are inspired by the system TACITUS. The answer is found by 'loose unification' of the question logical form and the answer logical form. If the unifiable answer cannot be found, the second large loop is repeated; i.e., after doing lexical alternations on the passage-retrieval query (synonyms, morphological derivations) the process of passage retrieval is repeated. The lexical alternations are done using WordNet; e.g., killed → killer → assassin.

Answer justification. The answer justification is done using theorem proving: A question has a theorem role, and knowledge facts and the answer are axioms.[1] The proving is done by backchaining from questions to axioms. There are three types of facts used as axioms: textual facts from the passage, world knowledge, and coreference resolution.

If the candidate answer cannot be justified (the query cannot be "proven"), the third large loop is repeated. Semantic alternations of the passage-retrieval query, based on WordNet, are done, and we go back to the passage retrieval. Example: definition of 'prefer' includes 'liking better', so 'prefer' is included in the query for question: Where do lobsters like to live?

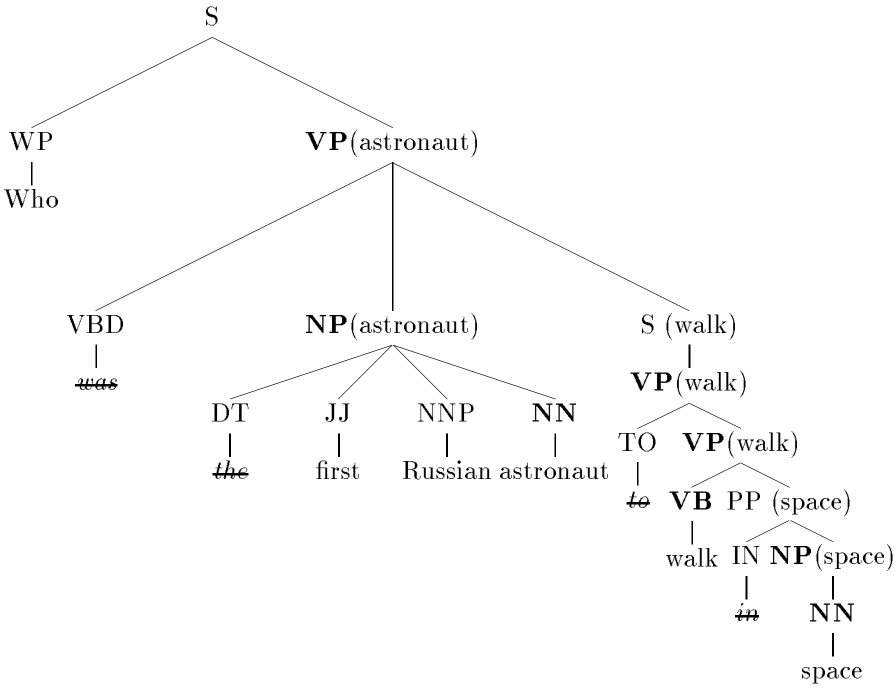
[1] About abductive reasoning: In deductive reasoning, we know the case and the rules, and infer the result. In inductive reasoning, we know the case and the result, and infer the rules. In abductive reasoning, we know the rules and the result, and infer the case.

5 Translating the FALCON logical form into a logical form based on AVMs

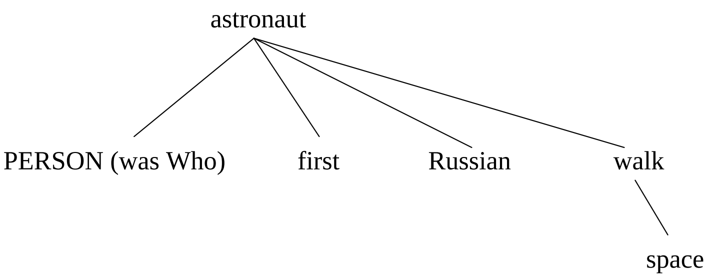
This is an example from from [HMP+00]. One of the questions used in TREC-9 QA is:

Who was the first Russian astronaut to walk in space?

The following is the parse tree. The stop words are in italics and stricken out, and the head daughters are in bold font. The node labels are shown in parentheses.



The labels are determined after ignoring stop words, and determining head daughters. The following graph representation of the logical form is obtained:



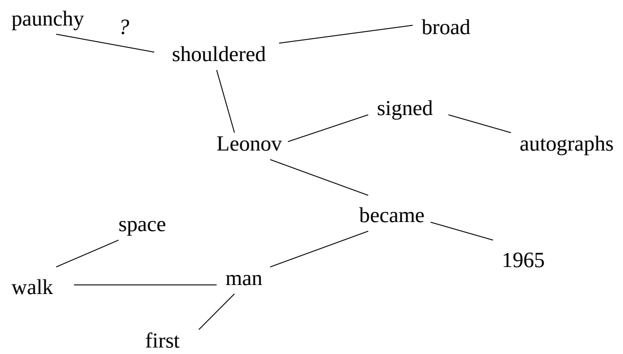
And, we get the logical form:

$$\text{astronaut}(x) \wedge \text{HUMAN}(x) \wedge \text{first}(x) \wedge \text{Russian}(x) \wedge \text{walk}(e, y, x) \wedge \text{space}(y)$$

Similarly, the sentence that contains an answer:

> The broad-shouldered but punchy Leonov, who in 1965 became the first man to walk in space, signed autographs.

is translated into the graph:



which is translated into the final logical form:

$$\text{paunchy}(y) \wedge \text{shouldered}(e_1, y, x)? \wedge \text{broad}(x?) \wedge \text{Leonov}(x) \wedge \text{HUMAN}(x) \wedge \text{signed}(e_4, v, x) \wedge \text{autographs}(v) \wedge \text{became}(e_3, z, u, x) \wedge 1965(u) \wedge \text{DATE}(u) \wedge \text{man}(z) \wedge \text{first}(z) \wedge \text{walk}(e_2, t, z) \wedge \text{space}(t)$$

The forms are taken from [HMP$^+$00], and the question marks are used for details that should be expressed in slightly different way, in my opinion.

How do we find an answer? First, we note that the node 'astronaut' in the question graph is the node with the greatest number of neighbours. So we assume that that node determines the target object, which is denoted by x in the logical form. After unification with a part of the answer logical form, we get a unification between x in the question form and z in the answer logical form.

Actually, the approach is not very successful in this case, since we would prefer a unification of the x in the question form to the x in the answer form. Nevertheless, this is the only case analyzed in [HMP$^+$00].

When translating the graph form to the logical form in this approach, which we call the classical logical form (LF), we first identify verb nodes. The other nodes (e.g., nouns) are translated into lexicalized predicates with one variable. If two such nodes are adjacent, then their predicates have the same argument. The verb nodes are translated into predicates that have one Davidsonian-event argument, and for each node neighbour there is one additional argument, which is a variable representing the adjacent node.

The advantage of this approach is its simplicity and elegance. One major drawback is that the produces semantic form is too much influenced by the syntax in the case of verbs. In the example, the verb 'walk' produces two unifiable predicates since it was used in the same syntactic way: "the first Russian astronaut to walk in space" and "the first man to walk in space". If another syntactic construction, like in the following sentence taken from the Web:

> The first space walk is made from the Soviet Voskhod 2 spacecraft by cosmonaut Alexei A. Leonov.

then the verb arguments might be sorted in different order, e.g.

$$\text{walk}(e, x, y) \wedge \text{space}(x) \wedge \text{Alexei A. Leonov}(x) \wedge \ldots$$

This example shows other problems with a simplistic approach, but we do not focus on them for now.

The problem can be solved by using named predicate arguments, like in attribute-value matrices (AVMs):

$$\begin{bmatrix} \textsf{EVENT:} & \text{walk} \\ \textsf{SUBJECT:} & \text{man} \\ \textsf{LOCATION:} & \text{space} \end{bmatrix}$$

Using AVMs, the previous question is translated into the following semantic representation:

$$\begin{bmatrix} \textsf{SELECT:} & \boxed{1} \\ \textsf{WHERE:} & \begin{bmatrix} \textsf{EVENT:} & \text{walk} \\ \textsf{SUBJ:} & \begin{bmatrix} \textit{astronaut} & \\ \textsf{NAME:} & \boxed{1} \\ \textsf{MOD:} & \text{first} \\ \textsf{MOD1:} & \text{Russian} \end{bmatrix} \\ \textsf{IN:} & \text{space} \end{bmatrix} \end{bmatrix}$$

A semantic representation of a part of the answer sentence is:

$$\begin{bmatrix} \textsf{EVENT:} & \text{walk} \\ \textsf{SUBJ:} & \begin{bmatrix} \textit{man} & \\ \textsf{NAME:} & \text{Leonov} \\ \textsf{MOD:} & \text{broad-shouldered} \\ \textsf{MOD1:} & \text{punchy} \\ \textsf{MOD2:} & \text{first} \end{bmatrix} \\ \textsf{IN:} & \text{space} \end{bmatrix}$$

If we assume that 'astronaut' is a subtype of 'man', and if we treat attributes 'MOD', 'MOD1', ...as one set-valued attribute, the two AVMs above can be unified.

References

[HMP+00] Sanda Harabagiu, Dan Moldovan, Marius Paşca, Rada Mihalcea, Mihai Surdeanu, Răzvan Bunescu, Roxana Gîrju, Vasile Rus, and Paul Morărescu. FALCON: Boosting knowledge for answer engines. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, 2000.

[TRE00] *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, 2000.

[VH00] Ellen M. Voorhees and Donna Harman. Overview of the ninth text retrieval conference (TREC-9). In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, 2000.