Question Answering over Unstructured Data without Domain Restrictions

Jochen L. Leidner
⟨leidner@linguit.com⟩

Linguit GmbH
Friedensstraße 10
76887 Bad Bergzabern
Germany.

University of Cambridge
Peterhouse
Cambridge CB2 1RD,
England, UK.

Abstract

Information needs are naturally represented as questions. Automatic Natural-Language Question Answering (NLQA) has only recently become a practical task on a larger scale and without domain constraints.

This paper gives a brief introduction to the field, its history and the impact of systematic evaluation competitions. It is then demonstrated that an NLQA system for English can be built and evaluated in a very short time using off-the-shelf parsers and thesauri. The system is based on Robust Minimal Recursion Semantics (RMRS) and is portable with respect to the parser used as a frontend. It applies atomic term unification supported by question classification and WordNet lookup for semantic similarity matching of parsed question representation and free text.

1 Introduction

Questions are the primary linguistic representation of a human information need. Automatic Natural-Language Question Answering (NLQA) is a task with a long history, but has only recently become practical on a larger scale and without domain constraints. Its advantages, once available, are obvious: easier, faster and cheaper access to information stored in databases or text documents, immediate feedback by virtue of automation, and eventually empowering the users without enforcing them to learn artificial query languages.

Computer users often use questions with conventional search engines, without realising that at the time of writing, they may not yet be equipped with NLQA abilities. For instance, if the users ask *When is the next train from Cambridge to London?*, a keyword-based engine will either ignore *'When'* as a stop-word or present a list of documents that are likely to contain (among others) the string *'When'*, in spite of the fact that this string is most unlikely to be contained on a page answering the question. Using NLQA systems, on the other hand, only pieces of information that refer to a time are returned.

For example, Table 1 shows an extract from the logfile of the *Magellan* search engine.[1] To sum up, *Natural Language Question Answering is the task of developing systems and techniques for making information access easier by using natural-language questions directly and effectively as search queries by exploiting their semantic structure*, without asking for additional human effort to translate the questions into sets of keywords or Boolean queries.

```
Where can I find free backrounds?
Where can I get information on diabetes?
does ureach.com use a pop3 mail server for
        email?
what is the 300th anniversary called?
where can I find plans for a small tool shed?
where can i find the lyrics of christina
        augular
How much liquid can a keg hold
What kind of dogs are good for people
        with allergies?
Where can I get buying advice for cars?
where can i find codes for driver for pc
```

Table 1: Some English questions, taken from the Magellan Internet search engine log (unedited).

The remainder of this paper is structured as follows: Section 2 very briefly describes the historical stages of NLQA.[2] Section 3 describes the general architecture of the implemented NLQA system for English and deals with the syntactic and semantic analysis and atomic term unification, the system's semantic similarity matching process. Section 5 describes question classification and named entity tagging. Section 6 then gives some experimental results. The paper concludes with a conclusion and outlook in Section 7.

2 Previous Work

The BASEBALL system by Green an co-workers (Green et al. (1969)) was a very early effort towards automatic question answering limited to the domain of baseball, and used a syntactic parser. In the 1970s, intensive research aimed to provide natural-language access to relational databases, but prior to the wide availability and global commercialisation of the Internet, databases

[1] At the time of writing, the Magellan search engine is no longer publicly available.

[2] For a more detailed account, also see the bibliography (Sabourin (1994)), which covers most works in the field until 1994.

to be questioned by end users were local and small, so menu-based access was sufficient for a small community of experts, and consequently no industrial impact was achieved.

More recently, the START (SynTactic Analysis using Reversible Transformations) NLQA system has been developed at MIT (Katz (1997)). It creates an offline knowledge base comprising normalised semantic tuples from English text and subsequently allows English queries being executed against that knowledge base.

The difference between the BASEBALL and START is that START is an *open-domain* question answering system: no domain limitation exists, although the type of questions that can be managed to get answered correctly is limited to factual questions (as opposed to expert question or in fact any type of query that involves more sophisticated 'deep' reasoning). Open-domain NLQA is significantly harder since no domain-specific assumptions or vocabulary can be hardwired in the system. BASEBALL, on the other hand, is a deep system relying on the *close-world assumption*, the tacit illusion of complete knowledge. Open-domain systems are shallow, based on models of pattern matching or Information Retrieval (IR); closed-domain systems typically use sophisticated logical deductions, but over an unrealistically tiny knowledge base.

Within IR, NLQA is to be contrasted with more conservative *document retrieval*, where the system would always reply with a ranked list of documents relevant to the topic characterised by the query rather than give a natural language reply directly, be it an extracted phrase or a generated reply.

In the 1990s, the broad availability of the WWW with its potential of 'information overload' and a series of evaluating competitions by US government organisations DARPA and NIST, namely the *Message Understanding Conferences* (Chinchor (1998)) and the *Text Retrieval Conference*'s question answering track (Voorhees and Tice (2000); Voorhees (2000)), have attracted more researchers back to the field, and the usual strategy adopted by participants is a mix between information retrieval and shallow linguistic filtering of retrieved result documents, so-called *passage retrieval*.

3 System Architecture

This section describes the design and implementation of **AskBill**, a simple NLQA system for English (Figure 1), in C++ in a very short time, namely eight person weeks of part-time effort.

Following the scenario of the TREC-8 NLQA track workshop, we take a set of questions as starting point. Every question had been prepared offline with some XML-like markup comprising three regions, namely the (tokenised) surface string of the question, a parse tree in LISP notation as generated by RASP (Carroll and Briscoe (2001)), and a set of functor-argument relations in *Minimal Recursion Semantics* (Copestake, Flickinger and Sag (1999)).[3]

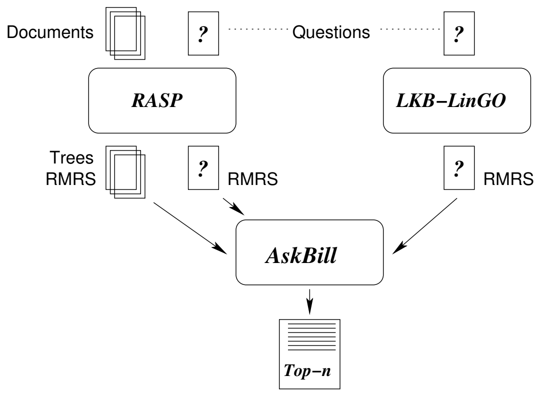

Figure 1: Architecture of the AskBill Question Answering System.

These semantic representations have been obtained

- using the LKB system with the public-domain LinGO resource grammar[4] and
- via a POS-based mapping described by template rules (e.g. 'map `NN1` to functors'), so as to yield, for instance `dog_NN1` $\rightarrow dog(e_j)$.

The document collection, on the other hand, had been prepared in the same fashion.[5]

AskBill reads a file with the already marked-up and parsed questions into main memory and iterates over all of them, creating `PossibleWorld` objects on the fly (Figure 2).

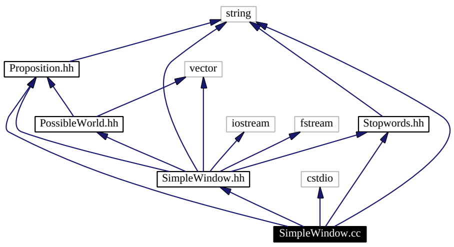

Figure 2: While traversing the input file, the class `SimpleWindow` creates `PossibleWorld` objects.

[3] Essentially, in RMRS any n-place predicate—such as $give(e_i, x, y, z)$—would be represented as a set of *elementary predicates*, $give(e_i)$, $arg_1(e_i, x)$, $arg2(e_i, y)$, $arg3(e_i, z)$, which simplifies incremental construction. See (Copestake (2002)) for an acessible, well-written tutorial using (R)MRS.

[4] ⟨`http://lingo.stanford.edu/`⟩

[5] With the exception that only RASP annotations are available, since a rule-based grammar such as LinGO fails on too many sentences to be useful.

For each question, a (possibly long) TREC document that is known to contain the answer is searched. For every question-answer pair $\langle q, a_k \rangle$, a score is computed using the method described below. For each question, a list of five best-scored answer candidates is kept and updated.

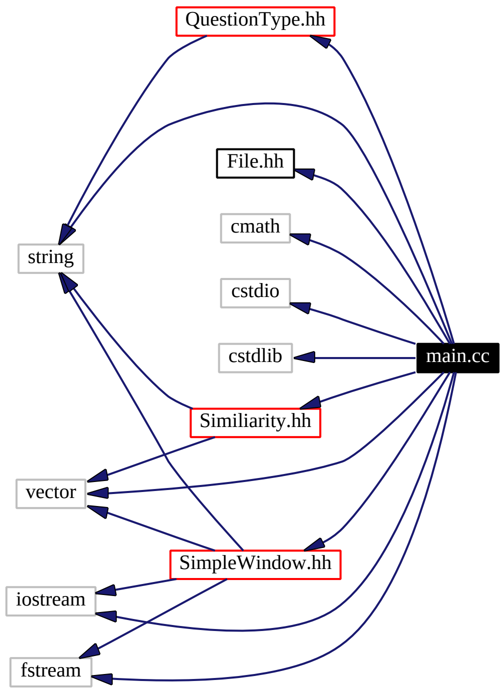

Figure 3: The dependency graph of the system (extract).

One of the advantages of the approach is *portability* across parsers: the parser is completely decoupled from the NLQA system proper, which simply works on the parsed questions and documents containing the answers, both in XML format.

4 Semantic-Based Scoring

Good results have been achieved with surface-based IR methods in Internet search engines, but such methods to break down when applied to retrieval of single paragraphs, sentences or even phrases. Since the TREC-8 data used has been annotated with functor-argument structures we can exploit these in the hope to improve precision by retrieving semantic relations rather than surface strings and recall by abstracting over word-forms and semantically equivalent expressions.

4.1 Matching Using Atomic Unification

Two semantic expressions $f(x)$ and $g(y)$ are equivalent if they have the same predicate name, $f = g$, and if they refer to the same things, $y = x$. In the following example, this becomes clearer: a set of *RMRS* (we also use the abbreviation of the theory–*Robust Minimal Recursion Semantics* for instances of its representations) can be considered a 'world', and we try to match question RMRS against candidate answers.

```
                  Q: PossibleWorld [
                       leave( e10 )
                       ARG1( e10 x11 )
                       Qintex( x11 )
                     ]

A1: PossibleWorld [            A2: PossibleWorld [
      leave( e222 )                  Qintex( x12 )
      ARG1( e222 x1 )                leave( e10 )
      Qintex( x1 )                   ARG1( e10 x11 )
    ]                              ]
```

Figure 4: Pairwise comparison of two (if microscopic) possible worlds A1 and A2, representing answer candidates, with a third (Q), representing a question's semantics.

In order to determine similiarity, we carry out a pairwise comparison and assign a *match score* for equal predicate names, equal arity, and equivalent arguments.[6] Because the argument numbering is idiosyncratic (as it depends on the number of phrases encountered so far) and usually does not correspond in any two possible worlds under consideration, we have to maintain a *binding list*. In effect, we perform *atomic term unification*. Atomic means here that arguments cannot in turn be terms, only variables and constants. Here is a trace of the above example:[7]

```
Scoreboard (score(Q, A1)=18): [
 5 3 0
 0 2 0
 0 3 5
] Bindings: { e10/e222 x11/x1 }

Scoreboard (score(Q, A2)=16): [
 5 0 3
 0 0 0
 0 5 3
] Bindings: { e10=e10 x11/x12 }
```

Optionally, function words can be ignored in this matching process, and predicate names may be stemmed (Porter (1980)).

4.2 Semantic Form: One Formalism, Multiple Parsers

Figure 5 compares the semantic representations as output by two parsers, namely the the robust shallow probabilistic parser RASP (Carroll and Briscoe (2001)) and the constraint-based unification parser LKB with the public-domain LinGO resource grammar for English (Copestake, Flickinger and Sag (1999)).

[6] These values can be controlled via command-line options.

[7] The scoreboard is simply a visual map that shows which parts contributed to the score; the score itself is a simple scalar (the sum of the values given).

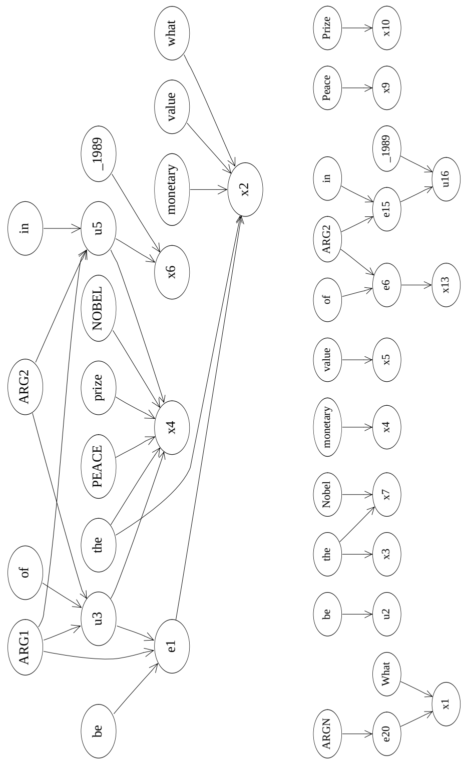

Figure 5: RMRS representations for the question *What was the monetary value of the Nobel Peace Prize in 1989?* as directed acyclic graphs: first LKB output using the LinGO grammar (left); robust RASP output (right).

Whereas the LKB analyses are superior in structural quality, due to ambiguity, there is a plethora of them, and the system currently always selects the first one in the absence of a probabilistic ranking. RASP, on the other hand, gives only one (the most likely) tree, but the mapping to semantic representations is still far from optimal.

5 Question Classification (QC)

Questions can be classified according to what is asked for, and this can help excluding unlikely candidates from the large set of potential answers. To this end, a flat classification was manually built from insights gathered whilst looking at the 200 test questions. Then, a set of regular patterns was devised to determine the categories. Using the publicly available pattern recognition engine *Regex++*, an automatic classifier was built. During runtime, question classes are then used to constrain the candidate answer set; Table 2 shows the resulting (flat) system of classes and shows how they relate to entity types.

Recognition of question types can be performed by pattern matching. Table 3 lists some instances that can easily be captured by regular expressions.

The follwing classes can be identified by AskBill:

- **TIME**: *yesterday, Dec 25, ...*

Question Type	Named Entities			
	TIME	LOC	PERSON	ORG
QSpacial		$\oplus$		
QTemporal	$\oplus$			
QPersonal			$\oplus$	
QCause				
QSpacialDistance		$\oplus$		
QSpacialExtension		$\oplus$		
QNumber				
QMonetary				
QTemporalDur	$\oplus$			
QTemporalFreq	$\oplus$			
QManner				
QOrganisational				$\oplus$

Table 2: Question types and recognised named entities

Name of Question types	How to Identify (Sample Pattern)
QSpacial	*Which* (+LOC) / *Where*
QOrganizational	*Which* (+ORG)
QPersonal	*Name* / *Who* / *Whom* (+ROLE)
QManner	*How*
QCause	*Why* / *What* (+REASON)
QNumber	*How many*
QMonetary	*How much* / *cost* / *rent* / *spend*
QTemporal	*When* / *(At) What time*
QTemporalDuration	*How long*
QTemporalFrequency	*How often*
QTemporalAge	*How old* / *At (Which / What) age*

Table 3: How to identify question types

- **LOC**: *Cambridge, Boston, ...*
- **PERSON**: *Mavis, Peter, Maya, ...*
- **ORG**: *Delta Airlines, JPL, ...*
- **TITLE**: *Mr, Dr., ...*[8]

Using a set of regular expressions, a simple Named Entity Recognizer was implemented. Some words, such as names for months are hardwired, and heavy use is made of cues such as *Mrs*, *PhD*, or *Mt*, and capitalisation. Here is an example:

> How many lives were lost in the China Airlines' crash in Nagoya, Japan?
> How tall is Mt. Everest?

5.1 Topic Detection and WordNet

Not all of the content words contribute equally to the determination of the topic of a question; to exploit this fact, a simple topic detector has been implemented that works on RASP parse trees. The module is helpful for determining the answer type of *what-* and *which*-questions, because their answer type cannot be discovered easily looking at the question word only. The detector extracts the first noun phrase in the question, discards all

[8] TITLE is useful in determining a PERSON.

```
Input tree:
[T
 [S What_DDQ
  [S be_VBZ
   [NP the_AT
    [N1 name_NN1
     [PP
      [P1 of_IO
       [NP the_AT
        [N1 [AP [A1 managing_JJ]]
                [N1 director_NN1]]]]]]]
            [PP [P1 of_IO [N1 Apricot_NN1
                           [N1 Computer_NN1]]]]]]?_?]
```

Result:
(managing) director

Figure 6: The topic detector at work (RASP tree notation simplified for presentation purposes)

words that are have neither nominal nor adjectival tags and marks the last noun as head (Figure 6). The words in the 'topic phrase', and especially the head, can subsequently be assigned a higher (doubled) match score.

We use the head noun of the topic phrase to determine the question types of the more difficult type *what/which*, for instance *In which year did the terrorist attack on WTC happen?* can be mapped to class QTemporal by fetching the hypernyms of *year*, possibly repeatedly, until we 'hit' a concept in Table 2 so as to allow to look up the corresponding question type. Here's another example which shows how the head noun *NATO* determines question type `QOrganisational`:

$$\begin{aligned} \texttt{NATO} \;\in\; & \{\texttt{NATO}; \texttt{world\_organization};\\ & \texttt{international\_organization}\}\\ \rightarrow\; & \{\texttt{alliance}; \texttt{coalition}; \texttt{alignment};\\ & \texttt{alinement}\}\\ \rightarrow\; & \{\underline{\textbf{organization}}; \texttt{organisation}\}\\ \rightarrow\; & \{\texttt{social\_group}\} \end{aligned}$$

If the question classifier is switched on, then the determined question type t_q will be compared against all answer candidate types t_{a_i}, and in the case of incompatibility, the respective candidates are discarded. The named entity tagger returns a number that compactly represents the presence/absence of named entity types in a_i (all t_{a_i} are OR-ed together). If the question type cannot be determined, no compatibility check is enforced. This mechanism can be implemented elegantly using bit vectors: a special encoding is used such that the bitwise AND of question type and complex entity type is non-zero in case of compatibility.

6 Evaluation

As in TREC-8, for every question, five answer candidates are output for evaluation. A ranked score is then computed using the TREC script provided by NIST (Voorhees (2000); Voorhees and Tice (2000)). A set of regular expressions was defined by NIST to automate the evaluation process, and a Perl script computes a score based on matching the expressions (describing the gold-standard answers) against the systems' retrieval results.

The scores used here are *Mean Reciprocal Rank (MRR)* and *Lenient Mean Reciprocal Rank (LMRR)*. Mean Reciprocal Rank assigns a value from the set {0, 0.2, 0.25, 0.33, 0.5, 1} to any 5-list of answer candidates, where the value chosen is $\frac{1}{r}$, where r is the rank of the answer that is considered the correct one by human judges. If no correct answer appears at any rank, the 5-list is scored zero. LMRR is defined here to get a better idea of the correctness of the answers; instead of assigning zero to unanswered questions, it ignores them and penalises only truly *wrong* answers. In any realistic scenario, no answer is always preferable to a wrong answer. The author has incorporated the additional computation of LMRR into the script. To make the scoring clearer, here is an example:

```
What nationality is Pope John Paul II ?
   1 1
   1 2
   1 3
   1 4
   1 5

What was the monetary value of the Nobel Peace
        Prize in 1989 ?
   2 1 The monetary value of the Nobel Peace
        Prize was $ 1000 ( USD ) in 1989 .
   2 2
   2 3
   2 4
   2 5

What debts did Qintex group leave ?
   3 1 I really wonder, too...and what about
       ENRON?
   3 2 Qintex group left debts of ADollars
       1.5 bn when it collapsed .
   3 3
   3 4
   3 5
```

The total number of questions is three, one of them unanswered (score 0), one has the correct answer at rank 1 (score 1) and another one has the correct answer at rank 2 (score $\frac{1}{2}$). This gives an MRR of $\frac{(0+1+\frac{1}{2})}{3} = 0.500$ and an LMRR of $\frac{(1+\frac{1}{2})}{2} = 0.750$.

It is interesting to note the heritage of the questions used here. Because TREC-8 was the first workshop to evaluate open-domain Q&A, a set of 127 questions made up by NIST specialists or supplied by the later participants of the competition was mixed with 49 questions from the human assessors (judges) and an existing logfile excerpt of 24 natural questions proposed to the FAQFinder system by undergraduate students playing with that system. (Voorhees and Tice (2000))

The allowable fragment size of text returned to the user turns out to be quite cruicial; a simple bag-of-words approaches can retrieve answers from paragraphs (250 character windows), but a 50-character restriction requires advanced NLP techniques to be competitive.

Description	Dataset	MRR	LMRR
RMRS, stop	RASP/TREC	0.463	0.764
RMRS, stop, QC	RASP/TREC	0.425	0.773

Table 4: Some Results for the Baseline System

Table 4 shows some preliminary results. Obtaining an MRR of 0.463 without any tuning, we can conclude that semantic (RMRS) matching with stop-word filtering appears to be robust and precise even without state-of-the-art question classification techniques. Large leading state-of-the-art research systems reach an MRR of 0.650 after several person-years of effort, although the results can of course not directly compared to the ones presented here due to the slightly different task definition (I evalue on sentences, not n-byte windows).

Adding the QC component decreases the MRR slightly by 3.8%, but increases the LMRR by 0.9%. This is in accordance with the usual precision-recall tradeoff, but it implies that whereas some wrong answers are discarded with an overall gain in correctness, some right answers were also discarded that would have been retrieved without QC. It can certainly be argued that the question answering task favours more precision over higher recall.

Further work will investigate more fine-grained question classification and incorporate a gazeteer component that has already been implemented.

7 Conclusion and Outlook

7.1 Conclusion

It has been shown how in a very limited period of time (eight person-weeks, part-time), a simple, but effective open-domain question answering system can be built from scratch using some off-the-shelf components, such as IR engines, POS taggers, and state-of-the-art parsers.

The result, AskBill, works on marked-up English full text that already contains POS-tags, syntactic parse trees, and semantic relations obtained by pre-processing using state-of-the-art systems. It can use[9] semantic-based matching and has been equipped with a simple named entity recogniser and question classifier. Of course the performance of such a small experimental system cannot compete with the best state-of-the-art systems, but it can still achieve a competitive MRR, which may partly be attributed to the the simplified scenario and the lacking separation between training and test data. Nevertheless, in a very short time a reasonable performance could be achieved.

7.2 Possible Enhancements

One disadvantage of the current approach is that the shapes of the trees are not exploited; early work by (Salton (1968)) has shown that parse tree similiarity can be utilised for matching in addition to predicate forms.

(Kay (1966)) describes a method for deriving valency/dependency graphs from parse trees, and such a method is expected to be exploited much more effectively to construct logical formulae than the currently employed, simple set of POS head-modifier mapping rules.

The most obvious extension, however, is not a linguistic one: the parser should be more tightly integrated, so as to allow interactive online deployment on top of an existing search engine such as *Google*. This would allow us to supplement the quantitative evaluation presented here with evidence of practical usefulness, and help gather a larger set of question to support further experiments.[10]

One elegant way of integrating a search engine is by means of a *Web Service*—essentially a remote procedure call mechanism that uses XML over HTTP—such as the Google API (Google (2002)).

7.3 Looking Ahead

Many topics in the realm of NLQA have not received much attention yet. This section will outline some key questions and issues for further research.

Authority. How is the authoritativeness of an answer to be judged? The publication source or original WWW site of the answer might help, if the publisher is a known authority in general or (even better) in the domain of the question (e.g. `nasa.gov` on astronomy). The redundancy of the Web should also be utilised, since multiple instances of the same answer can be taken as a signal for agreement.

Context. Often, a question asked presupposes additional implicit knowledge: in (1), the name of the president in question is most likely the one of the user's nation; in (2), *Berlin* is referentially underspecified with respect to Berlin, Germany and a multitude of Berlins in Northern America.

What's the name of the president? (1)

How many people live in Berlin? (2)

The user's extralingual context will have to be modelled using manual user profiles, his or her IP adress, login information, GPS data, previous questions asked, or other sources. If no such knowledge is available, a clarification dialogue may be triggered; in any case, the user should be made aware that alternative interpretations exist and should be able to access them on demand.

How is time-dependent knowledge to be handled? Consider the question

How many goals did Senegal score against France? (3)

At the time of writing, such a question would be taken to refer to the football World Cup 2002, and this contextual knowledge usually remains tacit common ground. The

[9] among other methods which are not described here

[10] The batch capability should be retained, however, since it is necessary for evaluation runs to allow objective testing in the face of the changing nature of the WWW.

date of the publication on the Web should be taken into account, and a windowing technique that weights the answer scores with a *most-recently first* score could make sure to give preference to the most recent sources.

How are anaphoric references treated? The referent must be mentioned in answer candidates in non-pronnomial form, i.e. we don't want (4)

Who is the US president? (4)

<u>He</u> was the president of the US. (5)

to be retrieved as an answer to the question (5). To a certain extend, this is already prevented by question classification. Given a reliable algorithm for resolving referents, we could improve sentence or paragraph scores by adding a bonus for sentences like (5) in the environment of a sentence that can function as an answer to a question of type `QPersonal`.

Multilinguality. A significant part of textual evidence left behind by humankind is in languages other than English. Many important regions are highly multilingual (e.g. Europe, to give just one example), and most large companies are multinational and therefore multilingual. Computer users also prefer their native language for computer interaction (6,7,8). These multilingual needs need to be catered for by making NLQA multilingual.

Qué está el tipo de cambio por la libra esterlina? (6)

What is the exchange rate for Pounds Sterling? (7)

Wie ist der Umrechnungskurs für britische Pfund? (8)

Presentation. How can the results be presented in the most user-friendly way? If there is just a single candidate answer sentence, we might just want to print that sentence; in fact, if the correct answer is among the top-five candidates, it is known to be at rank one in the majority of the cases. We could also emphasise the central NP that contains the answer type. The unit of output (paragraph, sentence, clause, phrase, word) could be made configurable (as a user preference), or set automatically, depending on an estimate for the answer candidate's correctness.

Wireless NLQA. Finally, NLQA systems should be able to work in the spoken medium to enable users to interact with them using mobile phones, and eventually personal communicators. Whereas in restricted domains this has already been possible for some years, wireless NLQA without domain restriction requires robust high-accuracy recognition with very large vocabulary. Special language models might be needed to cope with out-of-vocabulary words (Schofield and Kubin (2002)).

It is hoped that each of these lines for further research is followed up in the near future, and that natural language question answering will soon become as pervasive as telephone 'answer' machines are today.

Acknowledgements. The author would like to thank Ted Briscoe, Ann Copestake and Simone Teufel for suggesting the experimental scenario pursued here, for providing TREC data parsed form, and for their advice and feedback.

The author thankfully acknowledges the use of the Martin Porter's stemmer (C implementation). The Stop-word list used is derived from the one collected by Gerard Salton and co-workers as used for SMART. The Boost Library's Regex++ 3.0 library was written by John Maddock. The WordNet 1.7 software has been provided to the research community by Princeton University. All other components were built from scratch by the author using Standard C++.

The financial support of the following organisations is gratefully acknowledged: EPSRC (grant GR/N26579), The Cambridge European Trust, DAAD, Linguit GmbH and Peterhouse, Cambridge.

References

Abney, S., Collins, M. and A. Singhal (2000) 'Answer Extraction.' In: *Proceedings of the Applied Natural Language Processing Conference 2000 (ANLP'00).*

Beckwith, R., Miller, G. A. and R. Tengi (1993) 'Design and implementation of the WordNet lexical database and searching software.' [Online.]
⟨`http://www.cogsci.princeton.edu/~wn/`⟩

Burke, R. D., Hammond K., Kristian, J., Kulyukin, V. A., Lytinen, S., Tomuro, N., and S. Schoenberg (1997) *Question Answering from Frequently Asked Question Files: Experiences with the FAQ Finder System.* (Technical Report TR–97–05) Department of Computer Science, University of Chicago.

Carroll, J. and T. Briscoe (2001) 'High precision extraction of grammatical relations.' In: *Proc. of the 7th International Workshop on Parsing Technologies,* Beijing.

Chinchor, N. A. (1998) 'Overview of MUC-7/MET-2.' In: *Proceedings of MUC-7.* San Diego, CA: SAIC Corporation.

Copestake A. (2002) *Implementing Typed Feature Structure Grammars.* (CSLI Lecture Notes.) Chicago: Chicaco University Press.

Copestake A., Flickinger, D. and I. A. Sag (1999) 'Minimal Recursion Semantics–An introduction.' (draft manuscript) CSLI, Stanford University.

Fellbaum, C. (ed.) (1998) *WordNet: An Electronic Lexical Database.* Cambridge, MA: MIT Press.

Google, Inc. (2002) The Google Application Programming Interface (Google API) [Online.].
⟨`http://www.google.com/apis/`⟩

Green, B. F., Wolf, A. K., Chomsky, C. and K. Laughery (1963) 'Baseball: An automatic question answerer.' In: Feigenbaum, E. A. and J. Feldman (eds.) *Computers and Thought*, 207–216.

Harabagiu, S., Moldovan, D., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., Girju, R., Rus, V. and P. Morarescu (2001) 'The role of lexico-semantic feedback in open-domain textual question-answering.' *Proceedings of the 9th Text Retrieval Conference (TREC-9).*

Harabagiu, S. Pasca, M. and S. Maiorano (2000) 'Experiments with open-domain textual question answering.' *Proceedings of the 18th Conference on Computational Linguistics (COLING-2000)*, 292–298.

Hovy, E., Gerber, L., Hermjakob, U., Junk, M. and C.-Y. Lin (2001) 'Question Answering in Webclopedia.' In: *Ninth Text Retrieval Conference (TREC-9).*

Katz, B. (1997) 'From sentence processing to information access on the World Wide Web.' *AAAI Spring Symposium on Natural Language Processing for the World Wide Web.* Stanford University, Stanford. CA.

Kay, M. (1966) *The Tabular Parser: a Parsing Program for Phrase Structure and Dependency.* (Memo RM–4933–PR) RAND Corporation.

Miller G. A. (2002) *WordNet.* [Online.] ⟨http://www.cogsci.princeton.edu/\~wn/⟩

Moldovan, D., Harabagiu, S., Pasca, M., Mihalcea, R., Girju, R., Goodrum, R. and V. Rus (2000) 'The structure and performance of an open-domain question answering system.' In: *Proceedings of the Conference of the Association for Computational Linguistics (ACL-2000)*, 563–570.

Porter, M. F. (1980) 'An algorithm for suffix stripping.' *Program* **14***(3)*: 130–137. Reprinted in Spärck Jones, K. and P. Willet (eds.) (1997) *Readings in Information Retrieval.* San Francisco: Morgan Kaufmann.

Sabourin, C. F. (1994) *Natural Language Interfaces: Interfaces to Databases, to Expert Systems, to Robots, to Operating Systems, and Question-Answering Systems: Bibliography.* Infolingua, Montreal.

Salton, G. (1968) *Automatic Information Organization and Retrieval.* New York: McGraw-Hill.

Schofield, E. and G. Kubin (2002) 'On Interfaces for Mobile Information Retrieval.' In: *Proceedings of the Conference for Mobile Human-Computer Interaction 2002 (MHCI'02).*

Voorhees, E. M. and D. M. Tice (2000) 'Building a Question Answering Test Collection.' In: *23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,* Athens, 200–207.

Voorhees, E. M. (2000) 'The TREC-8 question answering track report.' In: *Proceedings of the Second international conference on Language Resources and Evaluation (LREC-2000).*

A System Output

```
leidner@tauris:~/> bin/askbill -stop -select 33
  -trace 0 -indiv trec8qa/rmrs data/ques.rmrs

AskBill - A simple open-domain question answering system
  Copyright (C)2002 by Jochen Leidner

using stop-word filter...
selected answering question 33
trace level set to 0...
using individual answer files...
system initialised... OK
200 questions read... OK

AskBill> What is the largest city in Germany ?

answering question 33
question 33: gunzip'ing ../trec8qa/rmrs/q33.rmrs.gz to
  /tmp/jl129.rmrs...

========================================
Scoring complete. Writing results...
========================================

33 1 The new Lander have to be completely rebuilt and
  there are enormous fortunes to be made by those willing
  to take risks on the future development of Berlin -
  with 3.4 m inhabitants Germany 's largest city and the
  future seat of government - and cities such as Leipzig
  and Dresden .
33 2 The 1991 election in Hamburg , Germany 's second
  largest city after Berlin , was won by the SPD , with
  an absolute majority of just one seat , but it could
  now face serious defections because of the disarray of
  the party at national level .
33 3 With 273,000 inhabitants it is the 20th largest
  city in Germany , bright and open .
33 4 Saxony , and particularly Leipzig , the state 's
  second largest city , was one of the centres of the
  printing and newspaper industry before the second
  world war , and the core of eastern Germany 's heavy
  industry under the former communists .
33 5 Qiao said Berlin , the largest city in Germany ,
  has great influence on the world .
```

Table 5: A demonstration run.