Natural Language Engineering **1** (1): 000–000. Printed in the United Kingdom

Natural Language Question Answering: The View from Here

L. HIRSCHMAN
The MITRE Corporation
Bedford, Massachusetts, USA

R. GAIZAUSKAS
Department of Computer Science, University of Sheffield,
Sheffield, U.K

(*Received 20 September 2001*)

1 Introduction: What is Question Answering?

As users struggle to navigate the wealth of on-line information now available, the need for automated question answering systems becomes more urgent. We need systems that allow a user to ask a question in everyday language and receive an answer quickly and succinctly, with sufficient context to validate the answer. Current search engines can return ranked lists of documents, but they do not deliver *answers* to the user

Question answering systems address this problem. Recent successes have been reported in a series of question-answering evaluations that started in 1999 as part of the Text Retrieval Conference (TREC). The best systems are now able to answer more than two thirds of factual questions in this evaluation.

The combination of user demand and promising results have stimulated international interest and activity in question answering. This special issue arises from an invitation to the research community to discuss the performance, requirements, uses, and challenges of question answering systems.

The papers in this issue cover a small part of the emerging research in question answering. Our introduction provides an overview of question answering as a research topic. The first article by Ellen Voorhees describes the history of the TREC question answering evaluations, the results and the associated evaluation methodology. The second paper by Buchholz and Daelemans explores the requirements for answering complex questions that have compound answers or multiple correct answers. The third paper by Lin and Pantel describes a new algorithm to capture paraphrases that allow a more accurate mapping from questions to potential answers. The fourth paper by Light, Mann, Riloff and Breck describes experiments that systematically factor and assess question answering into component subproblems.

The current state of the field is such that at best partial solutions can be provided to the broad challenges of question answering, and, given the limited space in this

issue, only some of these challenges can be considered here. Nevertheless we believe that the papers included in this issue are a significant reflection of the current state of achievement and the preoccupations of active researchers in the area. From the other submissions we received for this issue, it is clear that the field is in a phase of active system building and creative experimentation, and not so much one of reflective, comparative or theoretical analysis. Thus, while it might be desirable for an issue such as this to offer a consolidating, synthetic overview of progress to date and issues for the future, in reality all it can offer is a limited view from the ground level of an exciting, dynamic research area – "the view from here".

In the rest of this introduction we provide a brief discussion of the dimensions of question answering as a research area (section 2), followed by a pocket sketch of the history of natural language question answering (section 3), an overview of current approaches (section 4), a discussion of resources and evaluation methodologies (section 5), and we conclude with reflections on future directions for QA research (section 6). We hope that this introduction will provide a useful general perspective on question answering research which complements the detailed technical contributions of the other papers.

2 Question-Answering: Dimensions of the Problem

To answer a question, a system must analyse the question, perhaps in the context of some ongoing interaction; it must find one or more answers by consulting on-line resources; and it must present the answer to the user in some appropriate form, perhaps associated with justification or supporting materials.

Several recent conferences and workshops have focused on aspects of the question answering research area. Starting in 1999, the Text Retrieval Conference (TREC)[1] has sponsored a question-answering track which evaluates systems that answer factual questions by consulting the documents of the TREC corpus. A number of systems in this evaluation have successfully combined information retrieval and natural language processing techniques.

Evaluation using reading comprehension tests provides a different approach to question answering, based on a system's ability to answer questions about a specific reading passage. These are tests that are used to evaluate students' comprehension, and, as a result, they provide a basis for comparing system performance to human performance. This was the subject of a Johns Hopkins Summer Workshop[2] and a Workshop on Reading Comprehension at the ANLP-NAACL joint conference in Seattle in 2000 (Light et al.2000).

These conferences, workshops and evaluations are opening up the rich problem domain associated with question answering. This section provides an overview of some dimensions of this research in terms of:

- Applications

[1] See http://trec.nist.gov.
[2] See http://www.clsp.jhu.edu/ws2000/groups/reading/prj_desc.shtml.

- Users
- Question types
- Answer types
- Evaluation
- Presentation

2.1 Applications

Question answering has many applications (see section 3 for more discussion). We can subdivide these applications based on the source of the answers: structured data (databases), semi-structured data (for example, comment fields in databases) or free text (the focus of the articles in this volume). We can further distinguish among search over a fixed set of collections, as used in TREC (particularly useful for evaluation); search over the Web, as discussed in the Buchholz and Daelemans paper; search over a collection or book, e.g., an encyclopedia (Kupiec1993); or search over a single text, as done for reading comprehension evaluations.

We can also distinguish between domain-independent question answering systems and domain specific systems, such as help systems. We can even imagine applying question answering techniques to material in other modalities, such as annotated images or speech data. Overall, we would expect that as collections become larger and more heterogeneous, finding answers for questions in such collections will become harder – although the paper by Light, Mann, Riloff and Breck (this issue) indicates that having multiple answer sources (answer redundancy) increases the likelihood of finding an answer.

2.2 Users

Users can range from first time or casual users to repeat or "power" users who might use such a system routinely in the course of their work. Clearly, these different classes of users require different interfaces, ask different questions and want different kinds of answers. The issue of different users is discussed at length in a recent roadmap document for question answering research – see Burger et al. (2001). For first time users, it may be important to explain the limitations of the system, so that the user can understand how to interpret the answers returned. For expert users, it may be desirable to develop and update a model of the user, so that summaries can emphasize novel information and omit information previously provided to the user.

2.3 Questions

We do not yet understand how to predict what makes some questions harder than others. This is an issue of importance to the educational testing community, where testers must prepare and validate standardized tests such as reading comprehension tests (Kukich2000).

We can distinguish questions by answer type: factual answers vs. opinion vs.

summary. We focus here on questions with factual answers, although reading comprehension tests, for example, often include other kinds of questions (*What is this story about?* or *What is the author's attitude towards the main character in this story?*). The question answering roadmap (Burger et al.2001) includes tackling increasingly challenging kinds of questions in later years.

Next we can distinguish different kinds of questions: yes/no questions, "wh" questions (*who was the first president*, *how much does a killer whale weigh*), indirect requests (*I would like you to list ...*), and commands (*Name all the presidents...*). All of these should be treated as questions. However, systems that depend heavily on the use of "wh" words for clues (*who* needs a person answer, *when* needs a time answer) may have difficulty processing such questions when phrased as *name the first president* as opposed to *who was the first president*. The issue of detecting and learning paraphrases is the focus of the Lin and Pantel paper.

We have evidence that some kinds of questions are harder than others. For example, *why* and *how* questions tend to be more difficult, because they require understanding causality or instrumental relations, and these are typically expressed as clauses or separate sentences (Hirschman et al.1999). As the Light, Mann, Riloff and Breck paper discusses, if a system does a good job of analyzing the type of answer expected, this narrows the space of possible answers. Certain kinds of questions are harder to answer because of an insufficiently narrowed answer type; for example, *what* questions are notoriously hard, because they provide little constraint on the answer type (*what happened* vs. *what did they see* vs. *what did they do*).

2.4 Answers

Answers may be long or short, they may be lists or narrative. They may vary with intended use and intended user. For example, if a user wants justification, this requires a longer answer. But short answer reading comprehension tests require short answers (phrases).

There are also different methodologies for constructing an answer: through extraction – cutting and pasting snippets from the original document(s) containing the answer – or via generation. Where the answer is drawn from multiple sentences or multiple documents, the coherence of an extracted answer may be reduced, requiring generation to synthesize the pieces into a coherent whole.

In the limit, question answering and summarization may merge as research areas. A generic summary answers the question: what is this story about? And a topic-specific summary provides information in a story about the requested topic – in effect, an answer. See Mani et al. (To appear) for a discussion of intrinsic evaluation of summarization structured around providing answers to questions.

2.5 Evaluation

What makes an answer good? Is a good answer long, containing sufficient context to justify its selection as an answer? Context is useful if the system presents multiple candidate answers, because it allows the user to find a correct answer, even

when that answer is not the top ranked answer. However, in other cases, short answers may be better. The experiences of the TREC question answering evaluations (Voorhees, this issue) show that it is easier to provide longer segments that contain an embedded answer than shorter segments. In section 5, we discuss issues of evaluation and criteria for question selection and answer correctness in greater detail.

2.6 Presentation

Finally, in real information seeking situations, there is a user who interacts with a system in real time. The user often starts with a general (and underspecified) question, and the system provides feedback directly – or indirectly by returning too many documents. The user then narrows the search, thus engaging in a kind of dialogue with the system. Facilitating such dialogue interactions would likely increase both usability and user satisfaction. In addition, if interfaces were able to handle both speech input and dialogue, question answering systems could be used to provide conversational access to Web based information – an area of great commercial interest, particularly to telecommunications and Web content providers.

To date, there has been little work on interfaces for question answering. There have been few systematic evaluations of how to best present the information to the user, how many answers to present to a user, how much context to provide, or whether to provide complete answers vs. short answers with an attached summary or pointers, etc. This is an area that will receive increased attention as commercial question answering interfaces begin to be deployed.

3 A Brief History of Question Answering

There has been a dramatic surge in interest in natural language question answering since the introduction of the Question Answering track in the Text Retrieval Conferences, beginning with TREC-8 in 1999 (Voorhees and Harman2000). However this recent interest is by no means the first time the topic has been addressed by natural language processing (NLP) researchers. In fact, Simmons (1965) begins a survey article "Answering English Questions by Computer" with the statement that his paper reviews no fewer than *fifteen* implemented English language question-answering systems built over the preceding five years. These systems include conversational question answerers, front-ends to structured data repositories and systems which try to find answers to questions from text sources, such as encyclopedias.

3.1 Natural Language Front Ends to Databases

The best-known early question answering program[3] is BASEBALL (Green et al.1961), a program for answering questions about baseball games played in the American

[3] Defined here as taking as input an unrestricted range of questions in natural language, and attempting to supply an answer by searching stored data.

league over one season. Given a question such as *Who did the Red Sox lose to on July 5?* or *How many games did the Yankees play in July?* or even *On how many days in July did eight teams play?*, BASEBALL analysed the question, using linguistic knowledge, into a canonical form which was then used to generate a query against the structured database containing the baseball data.

While BASEBALL was relatively sophisticated, even by current standards, in how it dealt with the syntax and semantics of questions, it was limited in terms of its domain – baseball only – and by the fact that it was intended primarily as an interface to a structured database and not as an interface to a large text collection. In this regard BASEBALL was the first of a series of programs designed as "natural language front-ends to databases". In this tradition, the assumption was that computers would hold vast amounts of data in structured databases, the details of which would be opaque to many users. Rather than compel users – typically construed as time-pressured, computationally-challenged executives – to learn the structure of a database and a specialised language for querying it, the aim was to allow users to communicate in their own language with an interface that knew about questions and about the database structure and could negotiate the translation.

The most well-remembered other early work in this tradition is the LUNAR system. LUNAR was designed "to enable a lunar geologist to conveniently access, compare and evaluate the chemical analysis data on lunar rock and soil composition that was accumulating as a result of the Apollo moon mission" (Woods1973). LUNAR could answer questions such as *What is the average concentration of aluminum in high alkali rocks?* or *How many Brescias contain Olivine?*. More than a toy, it was demonstrated at a lunar science convention in 1971 and was able to answer 90% of the in-domain questions posed by working geologists, without prior instructions as to phrasing. Again note the limitation to a narrow domain.

Throughout the 1970's, further work continued in this tradition (see the articles on the PLANES, LADDER, and TEAM systems in Grosz et al. (1986)). A good review of this work through to 1990 can be found in Copestake and Sparck Jones (1990).

From the perspective of the current research focus in question answering, the key limitation of this work is that it presumes the knowledge the system is using to answer the question is a structured knowledge base in a limited domain, and not an open-ended collection of unstructured texts, the processing of which is itself a major part of the QA challenge. Of relevance, however, is the valuable work done in this area on the syntactic and semantic analysis of questions [4] and on the

[4] One interesting difference between the questions typically discussed in the literature on natural language front ends to databases and those in the literature on QA against open text collections is the role of quantifiers and logical connectives. In questions posed against databases, quantifiers and connectives frequently play a significant role – e.g. *Who are all the students in MAT201 who also take MAT216?*. Put otherwise, such questions tend to ask about the extensions of complex sets defined in terms of set theoretic operations on simpler sets. Questions against open text collections, on the other hand, tend to be about finding properties or relations of entities known via a definite description – *Where is the Taj Mahal?*, *What year did the Berlin Wall come down?*, *Which team won the FA cup in 1953?*. In such questions quantifiers and connectives do

pragmatics of the interchange between user and system – see, e.g. Webber (1986), for a discussion of the useful distinction between "answers" (the information literally requested by a question) and "responses" (which may include an answer but also possibly helpful information beyond what was requested, justification, clarification of misconceptions or mistaken presuppositions in the question, etc.) and arguments as to why answers alone are not enough for usable systems.

3.2 Dialogue Interactive Advisory Systems

While natural language front-ends to databases is an application area for question answering that attracted researchers early on, another area of initially purely theoretical interest was question answering in human-machine dialogue. As is well known, Alan Turing (1950) proposed conversational understanding as *the* test for machine intelligence; he presented his challenge in the form of an interrogator who poses questions to an unseen entity (person or machine) and is then asked to judge which is which on the basis of their responses.

Early dialogue systems such as SHRDLU (Winograd1972) and GUS (Bobrow et al.1977) were built as research systems to help researchers understand the issues involved in modelling human dialogue. SHRDLU was built for a toy domain of a simulated robot moving objects in a blocks world; GUS simulated a travel advisor and had access to a restricted database of information about airline flights. For both these systems, sample dialogues reveal the serious challenges that must be overcome in the building interactive advisory systems, particularly in dealing with anaphora and ellipsis.

Despite that fact that such early interactive question answering systems used structured data as their knowledge source there is no requirement that they do so – text collections could be used instead, though of course real-time response is essential for such systems. This is now the focus of current international research on conversational spoken language interfaces. For example, MIT's Jupiter system provides a telephone-based conversational interface for international weather information (Zue et al.2000)[5]. It harvests on-line weather information from multiple web sites, and responds to naturally phrased questions, such as *What will the weather be tomorrow in Tokyo?*. Such systems point out the many user-centered and pragmatic issues in question answering that are easy to overlook if one is focused solely on the ability to express complex queries and get correct responses to them from complex data sets.

3.3 Question Answering and Story Comprehension

An obvious way to test whether someone has understood a text is to ask them questions about it: if they can answer correctly, they have understood; if not, they

not play a major role. No doubt this will change as open text collection QA gets more ambitious, bringing these two traditions closer together.

[5] See also: http://www.sls.lcs.mit.edu/sls/whatwedo/applications/jupiter.html.

have not. This technique is widely used for testing humans – e.g. for determining reading levels of children or second language learners – and early on was recognised as an appropriate way of testing the capabilities of natural language understanding systems.

The most notable early work here is that of Wendy Lehnert. Working within Schank's framework of scripts and plans as devices for modelling human story comprehension (Schank and Abelson1977), she devised a theory of question answering and an implementation of that theory in a system called QUALM (Lehnert1977). Her key concern in this work was to move away from the view that natural language question answering should be seen merely as a front-end to a completely separate data or information retrieval process. Instead she viewed the process of question answering as one in which both the understanding and answering of a question relies on the context of the story and pragmatic notions of appropriateness of answer. In her approach both question and story text are analysed into a conceptual dependency representation. But question answering is not just a process of matching these representations. The interpretation of a question also requires it to be assigned one of thirteen conceptual categories, such as "Verification", "Request", "Causal Antecedent", "Enablement", "Instrumental/Procedural", etc. Classifying questions this way is necessary to avoid answering questions such as *Do you know the time?* with *Yes* or *How did John pass the exam?* with *A pen.* Further inference may need to be made on the basis of context. To answer a question such as *Who wasn't at the Math lecture today?* an exhaustive list of most of the world's population is not required. Once the question is interpreted, answering may still require more than simply matching against memory. Expectations that stories may have aroused when told, and then contradicted, may need to be recreated to answer a question. On being told John ordered a hamburger we may assume he ate it. But if the story goes on to say it was so burnt he left the restaurant, we cancel that assumption. However, the literal representation of the story will not contain the fact that the hamburger was not eaten. If the question is then posed *Why did John not eat the hamburger?* then failing to provide any answer is not a good response, and not one a human would make. The appropriate response is that it was burnt. To make this response may require recreating expectations at answer retrieval time and determining what in the text violated them. The key point is that comprehension, as tested by this kind of question, is a dynamic process that involves integrating world knowledge and the information literally conveyed in the text.

Further work has gone on in story comprehension, but much of it within the psychology community (see, e.g, Kintsch (1998)) and work on developing computational models of story understanding dwindled through the 1980's and 1990's. However, there has been a recent revival of interest in the area, following the creation of a reading comprehension evaluation task (Hirschman et al.1999). Arguably, the area was held back because there was no agreed way to evaluate systems and evaluation has assumed a much more central role in the methodology of natural language research over the past fifteen years. But reading comprehension tests offer a solution to this problem, a solution which has the additional merit of being inex-

pensive, in that test materials are already available for humans and do not require special efforts to produce them.

The story comprehension work shares with current open text collection QA the characteristic that answers to questions must be derived from unstructured texts. However, like natural language front ends to databases and advisory systems, the questions asked to a story comprehension QA system may follow on from each other in a dialogue-like way, and may involve anaphora or ellipsis between questions (*Who was US president in 1958? In 1960? ... Which party did he lead?*). Further, unlike open text collection QA, the text containing the answer is known in advance. Multiple questions about a single text force a deeper processing of that text, and the problems of noise introduced by similar but irrelevant texts are avoided, as are the computational issues surrounding the processing of massive numbers of texts. However, story comprehension tests tend to provide less answer redundancy, which increases the difficulty of the answer location task, as discussed in the Light, Mann, Riloff and Breck paper.

3.4 Information Retrieval, Information Extraction and Question Answering

Information retrieval (IR), which, following convention, we take to be the retrieval of relevant documents in response to a user query, has been an active research area since the mid-1950's (Sparck Jones and Willett1997). It is related to question answering in the sense that users form queries because they wish to find answers to questions. However, beyond this the similarity largely ends. IR systems return documents, not answers, and users are left to extract answers from the documents themselves. Furthermore, the queries users put to IR systems need not be formed as syntactically correct interrogatives, and in fact may suffer for being so. And, subtle syntactic differences, as, for example, between the questions *Who killed Lee Harvey Oswald?* and *Who did Lee Harvey Oswald kill?*, are completely lost on most IR systems, which simply reduce a query to a bag of stemmed open class words.

IR is, however, relevant to question answering for two reasons. First, IR techniques have been extended to return not just relevant documents, but relevant passages within documents. The size of these passages can be steadily reduced, at least in theory, so that in the limiting case, what is extracted is, effectively, just the answer to a question. Thus, question answering can be thought of as passage retrieval in the limit. Second, the IR community has, over the years, developed an extremely thorough methodology for evaluation, the most well-known current exemplars of which are the annual Text REtrieval Conferences, or TRECs, run by the US National Institute of Standards and Technology. It is from this methodology and community that the recent question answering evaluation developed, which in turn has stimulated much of the current interest in question answering (Voorhees, this issue).

The other strand of research that has fed into the current TREC question answering track is information extraction (IE) or, as it was initially known, message understanding. IE can be defined as the activity of filling predefined templates from

natural language texts, where the templates are designed to capture information about key role players in stereotypical events (Gaizauskas and Wilks1998). For example, a template is easily defined to capture information about corporate take-over events; such a template has slots for the acquiring company, the acquired company, the date of the acquisition, the amount paid, etc. Running an IE system designed to fill this template over large volumes of text results in a structured database of information about corporate take-overs. This database can then be used for other purposes, e.g., database queries, data mining, summarisation. In the current context, IE templates can be viewed as expressing a question and a filled template as containing an answer. Thus, IE may be viewed as a limited form of question answering in which the questions (templates) are static and the data from which the questions are to be answered are an arbitrarily large dynamic collection of texts.

The IE community devised its own evaluation exercise – the Message Understanding Conferences, or MUCs [6]– which ran between 1987 and 1998. The termination of the MUC exercises, coupled with the desire to continue to push language understanding technology in novel directions via open evaluation exercises, were enabling conditions for the current TREC question answering evaluation.

3.5 The Logic of Questions and Answers

The preceding sections have presented a sketchy survey of work on automated question answering. There is also a body of work outside computer science on questions and answers, some of which is of relevance and has influenced work on automated question answering.

In 1955 M.L and A.N. Prior (1955) introduced the term "erotetic logic" to refer to the study of questions as logical entities distinct from statements. While this study by no means started with them (see, e.g. Hamblin (1967) for references to philosophical work on questions in Aristotle, medieval logic, and the 19th-century), their work is an early expression of what became, over the next 20 years, a serious attempt to apply formal logical techniques to the analysis of questions; i.e., to define a suitable syntax and semantics for a formal language of questions and answers.

As with parallel efforts in the logic of assertion, the efforts of logicians working in this area have not primarily been directed towards accounting for natural language usage. Rather, they have been concerned with providing a good formal notation and set of conceptual distinctions for investigating questions and answers:

> We hope thus to illuminate the question-answer situation in English in much the same way as formal logic illuminates the inference situation in English, in order to thereby contribute to our understanding of the erotetic "deep structure" of natural language. (from the Introduction to Belnap and Steel (1976)).

A good review of work on the logic of questions and answers can be found in Harrah (1984). He covers in particular depth the work of Belnap and Steel (1976) and

[6] Proceedings of the last MUC are available on-line at: http://www.itl.nist.gov/iaui/-894.02/related_projects/muc/index.html .

of Åqvist (1975). Belnap and Steel's account provides a good example of the general concerns of logicians working in this area. They begin by assuming a clearly defined assertoric language (first order predicate calculus with functions and identity). They then formally define the key basic notions of *elementary questions* – either whether-questions or which-questions – and *direct answer* and argue that these formal definitions capture essential intuitions about basic questions and answers to them. Given this basis they go on to explore more complex forms of questions and answers and notions of presupposition, effectiveness and completeness. The result is a very rich formal account which provides a set of analytical tools of considerable potential utility for automated question answering.

Of course logicians are not interested in *how* answers to questions are derived in practice. There is a strand of work on computing answers to logical queries posed against logic databases. This tradition starts with Green's work (Green1969) on using resolution theorem provers to capture the instance found in constructing a proof of an existentially quantified formula, and leads on into logic programming and deductive databases. However, this work captures a very small part of the question-answer logic developed by logicians such as Belnap and Steel (1976). One of the challenges facing researchers into natural language question answering is how to bridge, or at least narrow, the gap between engineering experimentation and theoretical understanding. Both sides will benefit from work of the other.

4 Overview of Current Approaches

The previous section has indicated the scope of work relevant to the general task of automated question answering. Let us now look at the sorts of approaches which are currently being employed to address this task.

As a framework for discussing actual systems, it is useful to have in mind a generic architecture for the QA task. Specific systems can then be seen as instantiations of the general architecture, with particular choices being made concerning representation and processing for each component of the overall model.

Figure 1 proposes such a general architecture for the QA task, conceived as that of asking natural language questions to a system that has as its knowledge source a large collection of natural language texts. Not all QA systems will implement all components in the model (in particular most current TREC QA systems do not utilise dialogue or user models); and, there may well be systems that implement functionality not in the model, or which cannot be easily mapped into it. Still, having such a general model in mind is useful, and helps to guide and structure discussion.

We briefly describe each of the processing stages in the model, then return to each stage in somewhat more detail, discussing issues to be faced when implementing that stage and exemplifying choices by reference to actual systems – for the most part systems developed to participate in the TREC Question Answering Track. However, systems developed for, e.g., the reading comprehension task can also be described in these terms.

1. *Question Analysis* The natural language question input by the user needs to

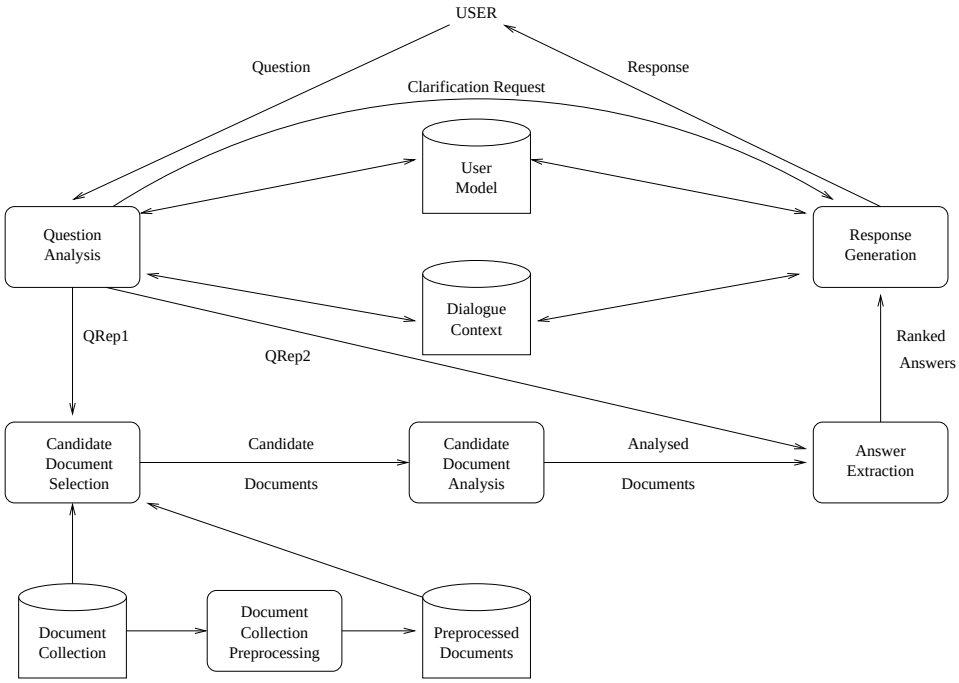

Fig. 1. Generic Architecture for a Question Answering System

be analysed into whatever form or forms are needed by subsequent parts of the system. The question may be interpreted in the context of an on-going dialogue and in the light of a model which the system has of the user. The user could be asked to clarify his or her question before proceeding.

2. *Document Collection Preprocessing* Assuming the system has access to a large document collection as a knowledge resource for answering questions, this collection may need to be processed before querying, in order to transform it into a form which is appropriate for real-time question answering.
3. *Candidate Document Selection* A subset of documents from the total document collection (typically several orders of magnitude smaller) is selected, comprising those documents deemed most likely to contain an answer to the question.
4. *Candidate Document Analysis* If the preprocessing stage has only superficially analysed the documents in the document collection, then additional detailed analysis of the candidates selected at the preceding stage may be carried out.
5. *Answer Extraction* Using the appropriate representation of the question and of each candidate document, candidate answers are extracted from the documents and ranked in terms of probable correctness.
6. *Response Generation* A response is returned to the user. This may be affected by the dialogue context and user model, if present, and may in turn lead to their being updated.

4.1 Question Analysis

The first stage is question analysis. The input to this stage is, by assumption, a natural language question, though this needs qualifying in several ways. First, there may be constraints on the input language. For example, the user may be required to use a subset of natural language, a "controlled language", which is limited in terms of vocabulary and syntax – most natural language front ends to databases (section 3.1) are limited in this way. It may even be that the user is constrained to use a form-filling interface for expressing questions that significantly simplifies the system's task of interpreting the question, albeit limiting the expressivity available to the user (see the Buchholz and Daelemans paper in this issue). Second, in addition to the explicit input of the question string there may be implicit input, in the form of context, if the system supports an on-going dialogue (so, e.g. there may be ellipsis or anaphora in the question which requires access to dialogue context to be interpreted). Other implicit input could be the system's knowledge of the user and his or her goals.

Output from this stage is one or more representations of the question for use in subsequent stages. For example, if the candidate document selection mechanism to be used in the next stage is an IR system, then one question representation might be a stemmed, weighted term vector for input to the search engine. However, this representation is unlikely to be adequate to allow exact answer strings to be picked out of the documents returned by the search engine. To do this, most systems resort to more detailed analysis of the question which typically involves two steps:

1. identifying the semantic type of the entity sought by the question (a date, a person, a company, and so on);
2. determining additional constraints on the answer entity by, for example:
   (a) identifying key words in the question which will be used in matching candidate answer-bearing sentences; or,
   (b) identifying relations – syntactic or semantic – that ought to hold between a candidate answer entity and other entities or events mentioned in the question.

The first step requires first looking at the key question word – *when* seeks a date or time; *where* a location; *who* a person. However, this is not enough, since various English question words, such as *which* and *what* do not carry much semantic typing information. The type of entity questions such as *Which company ...?* or *What building ...?* are seeking is also easy to determine. But for questions that involve more syntactically complex constructions such as *What was the Beatles' first hit single?* or *How many first class degrees in Computer Science were awarded at Cambridge last year?* things become more difficult.

Various systems have, therefore, built hierarchies of question types based on the types of answer sought, and attempt to place the input question into the appropriate category in the hierarchy. Moldovan et al. (2000), for example, manually constructed a question type hierarchy of about 25 types from the analysis of the TREC-8 training data. Srihari and Li (2000) base their question type hierarchy on

an extension of the MUC named entity classes and use a shallow parser to identify the question type, or what they call the *asking point.* Hovy et al. (2001) constructed a QA typology of 47 categories based on an analysis of some 17,000 "real" questions, extending the analysis to look beyond the semantic type literally requested so as to classify questions like *Who discovered America?* as Person, while classifying questions such as *Who was Christopher Columbus?* as Why-Famous. Harabagiu et al. (2001) describe a manually crafted top-level answer type hierarchy which links into parts of WordNet to extend the set of possible answer types available to their system.

Once the type of entity being sought has been identified, the remaining task of question analysis is to identify additional constraints that entities matching the type description must also meet. This process may be as simple as extracting keywords from the rest of the question to be used in matching against candidate answer-bearing sentences. This set of keywords may then be expanded, using synonyms and/or morphological variants (Srihari and Li2000) or using full-blown query expansion techniques by, e.g. issuing a query based on the keywords against an encyclopedia and using top ranked retrieved passages to expand the keyword set (Ittycheriah et al.2001). Or, the constraint identification process may involve parsing the question with grammars of varying sophistication. Harabagiu et al. (2001) use a wide-coverage statistical parser which aims to produce full parses. The constituent analysis of a question that it produces is transformed into a semantic representation which captures dependencies between terms in the question. Scott and Gaizauskas (2001) use a robust partial parser which aims to determine grammatical relations in the question where it can (e.g. main verb plus logical subjects and objects). Where these relations link to the entity identified as the sought entity, they are passed on as constraints to be taken into account during answer extraction.

4.2 Document Collection Preprocessing

If questions are to be answered in real time against gigabytes, and soon, terabytes, of text then off-line preprocessing of the text is necessary. So far most TREC QA systems appear to rely on conventional document indexing engines to do this. However, there is certainly no need to limit preprocessing to this sort of term indexing. Even if the candidate document selection stage relies on a conventional search engine to make its first selection of documents, having prestored a more extensive analysis of all texts in the text collection would render the candidate document analysis stage of our generic model unnecessary. For example, if one's system relied on building a logical form meaning representation of a text before attempting to extract answers from it, there is no reason in principle why this processing cannot be done in advance for the whole text collection. One system that adopts this approach is the ExtrAns system (Molla Aliod et al.1998) which derives logical representations of the document collection in advance of any querying. A system that does shallow linguistic processing of the document collection in advance is the SRI Highlight Information Extraction system (Milward and Thomas2000). This system does tagging, named entity recognition and chunking over large document sets

off-line and then stores the results as indexed constraints that can be matched during real-time user interaction with the system. Prager (2001) preprocesses the document collection and annotates terms with one of 50 semantic tags, which are indexed during the document indexing process in addition to the terms themselves. Katz (1997) extracts ternary relation expressions of the form `<subject relation object>` from syntactic analyses of natural language sentences in Web pages and builds an indexed database from them to support subsequent question answering against the Web.

4.3 Candidate Answer Document Selection

As observed, most existing TREC QA systems use some form of conventional IR search engine to select an initial set of candidate answer-bearing documents from a large text collection. Choosing this approach to winnow down the overall collection to a much smaller set of documents to be examined in detail is not the end of the matter, however. First, one must decide whether one wants to use a boolean or ranked answer search engine. Despite the higher results of ranked answer engines in standard IR evaluation, certain TREC QA participants have argued that boolean engines are more suitable for use in conjunction with a QA system (Moldovan et al.2000). If a ranked answer engine is used, a decision must be made as to how many retrieved documents will be used, i.e. how far down the ranking to consider; if a boolean engine is used, the issue of restricting the number of returned documents to examine still needs to be addressed. Second, the search engine may allow passage retrieval, and various parameters need setting here (passage length, passage windowing interval). Or, subsequent to retrieval, a topic-based text segmenter may be used to identify coherent text segments shorter than a full document which may then be re-ranked. See, for example, Clarke et al. (2001) who present and evaluate an algorithm for passage selection specifically for question answering and Hovy et al. (2001) for some experiments with how far down a ranked segmentation list to proceed. Prager (2001) investigates the question of whether combining results from multiple search engines can improve performance, and concludes that it can, at least to a limited extent.

4.4 Candidate Answer Document Analysis

Once candidate answer-bearing documents or document passages/segments have been selected, these text segments may then be further analysed. This will not be necessary if the system has already fully preprocessed all documents (as discussed above in 4.2) or if it is not designed to perform any further analysis.

Typically, however, systems now analyse the selected documents or document portions using at the very least a named entity identifier, which recognises and classifies multiword strings as names of companies, persons, locations, etc. The classes of names which are identified tend minimally to be those defined in the

Message Understanding Conference Named Entity Task[7], but in many cases these have been extended to include a variety of additional classes, such as products, addresses and measures, or refined to include subclasses, such as towns, cities, provinces, and countries.

Typical also at this stage are sentence splitting, part-of-speech tagging, and chunk parsing (identifying noun groups, verb groups, some prepositional phrases, etc.). Ferret et al. (2001), for example, describe a QA system which uses shallow syntactic analysis to identify multiword terms and their variants in the selected documents and to reindex and re-rank the documents before matching against the question representation. Some systems go further and do a fuller syntactic analysis followed by some sort of transduction of the derived syntactic structure into a set of relational constraints expressed either in a logical language or using relational labels between selected terms in the original sentence (e.g. between chunk heads). So, as noted above in the discussion of question analysis, Harabagiu et al. (2001) employ a wide-coverage statistical parser trained on the Penn Treebank to derive a dependency representation of sentences in the candidate answer documents, and then map this dependency representation into a first order logical representation, as they have done with the question; other QA systems that employ syntactic analysis to map candidate answer bearing documents into a logical or quasi-logical form prior to answer extraction are described by Molla and Hess (1998), Scott and Gaizauskas (2001), Zajac (2001). Hovy et al. (2001) also use a parser trained on the Penn Treebank (see Hermjacob (2001)), but in their case, rather than deriving syntactically-oriented phrase structure tree, and then mapping this into a logical form representation, they instead derive a representation of the sentence directly annotated with semantic role information; Buchholz and Daelemans (this issue) describe a "grammatical relation finder", again trained on the Penn Treebank, which adds relational labels such as subject and object between NP and VP chunks previously found by a chunker.

4.5 Answer Extraction

At this stage the representation of the question and the representation of the candidate answer-bearing texts are matched against each other and a set of candidate answers is produced, ranked according to likelihood of correctness.

Typically, systems that have analysed the question into an expected answer type plus, optionally, some set of additional constraints will also have analysed the candidate documents, or document segments, at least as far as annotation with semantic types drawn from the set of answer types. Thus, the matching process may require first that a text unit from a candidate answer text (perhaps a sentence, if sentence splitting has been carried out) contain a string whose semantic type matches that of the expected answer. Matching here can be type subsumption – perhaps construed

[7] The MUC-7 Named Entity task definition is available from: http://www.itl.nist.gov/-iaui/894.02/related_projects/muc/index.html .

as hyponomy in a lexical resource such as WordNet – and need not be restricted to identity.

Then, once a text unit containing an expected answer type has been found, other constraints may be applied to the text unit. These constraints may be viewed as absolute, so that failure to satisfy them rules out the candidate; or they may be viewed as preferences, which can be used to assign a score to the candidate for use in ranking the answer. Considerable variation between systems exists in terms of the types of constraints used at this stage, how constraint satisfaction is carried out, and how constraints are weighted.

For example, Moldovan et al. (2000) follow this approach. Once an expression of the correct answer type is found in a candidate answer-bearing paragraph, an answer window around the candidate is established and various quantitative features such as word overlap between the question and the answer window are used in a weighted numerical heuristic to compute an overall score for the window. Thus, for every candidate answer-bearing paragraph which contains an expression of the correct answer type, a score is derived for the answer-window containing the answer candidate and these scores are used to compute an overall ranking for all answer candidates. Harabagiu et al. (2001) extends this approach by using a machine learning algorithm to optimise the weights in the linear scoring function which combines the features characterising the answer windows.

Srihari and Li (2000), reverse the order of this general procedure, first applying question constraints other than expected answer type to rank sentences in candidate answer-bearing text segments and then using the expected answer type as a filter to extract the appropriate portion (e.g. 50 bytes for TREC) of the selected sentences. To rank sentences they use features such as how many unique question keywords are found in the sentence, the order of keywords in the sentence compared to their order in the question, and the whether the key verb or a variant matches. Ittycheriah et al. (2001) combine both expected answer type matching and a variety of word-based comparison measures in a single scoring function which they apply to three sentence windows which they move over candidate answer-bearing documents. See Light et al. (this issue) for a discussion of upper bounds on word-based comparison approaches.

Systems which derive richer document and question representations, i.e. logical forms or text annotated with semantic or grammatical role information, can use the additional constraints expressed in these representations to constrain the matching process. For example, a system that can identify logical subjects and objects can correctly identify *Jack Ruby* as the answer to the question *Who killed Lee Harvey Oswald?* when presented with the sentences *Ruby killed Oswald* and *Oswald killed Kennedy* – something that word overlap approaches have difficulties with. However, most systems which utilise such grammatical constraints have realised that for the system to be robust the constraints must be treated as preferences only, and not as mandatory. For while when they match they are likely to guarantee a correct answer, to insist that they match is to demand too much – sacrificing too much recall for precision. Thus, systems that can use semantic or grammatical role information typically fall through to less principled word overlap measures when their principled

constraints do not get instantiated. Hovy et al. (2001), Scott and Gaizauskas (2001), and Buchholz and Daelemans (this issue) are all examples of systems which exploit richer document representations where possible, but have a fallback strategy when the richer constraints are not applicable.

4.6 Response Generation

For the TREC QA evaluations, the sole response that most systems generate is a ranked list of the top five answers, where each answer is a text string [8] of up to n bytes (where $n = 50$ or $n = 250$) which has been extracted from a text (or texts) in the document collection.

This sort of response is likely to be inadequate in real applications for a variety of reasons. First, n-byte extracts are unlikely to be grammatical, or make good reading. Minimally, they need links back to their source documents to provide linguistic context (e.g. to resolve dangling anaphors). Or they need to be rephrased so as to make them comprehensible. Second, users may want more or less evidence or context for the answer. This will depend on the user and how much they trust the system and how much they know already about the topic in question. Third, answers may be more complex or extensive than users had anticipated, and the system may need to make decisions to truncate its output, or to initiate a dialogue with the user in order to decide how to proceed. See Buchholz and Daelemans (this issue) for a discussion of complex and answers and a different approach to presentation of search results.

5 Resources and Evaluation

This section addresses two critical issues for the development of question answering as a research area: resources and evaluation. These two issues are closely intertwined; developers need resources into order to build systems and they need evaluation methods (and training and test data) in order to evaluate the effectiveness of their systems. Historically, the introduction of large-scale common evaluations, such as TREC, has created strong communities of interest and has accelerated research progress. In this section, we look at some of the specialised resources needed for question answering. We then discuss evaluation, with a particular focus on methods for automated evaluation.

5.1 Resources

To create a question answering system, researchers need corpora of question-and-answer sets. Ideally, these sets would be naturally occurring questions, whose answers are contained in (or derived from) some larger collection of documents.

Kupiec (1993) used trivial pursuit questions as a source of question-answer pairs,

[8] Answers may also be list of strings for the so-called "list" questions like *Name three waterfalls over 100 meters* introduced in TREC-10.

and an on-line encyclopedia as the "collection" in which to search for answers. Recently, the TREC question answering track has been a valuable source of question/answer pairs occurring in collections of news stories. For reading comprehension, there are now several corpora of short answer reading comprehension tests available; see Light, Mann, Riloff and Breck, this volume, and also (Hirschman et al.1999).

However, to use machine learning and statistical techniques effectively, larger corpora are needed. For example, questions have a syntax that is different from assertions, and a large corpus of expository or narrative prose will typically contain very few questions – so that the rules developed on the basis of general corpora will not work well (without specialised tuning) for question analysis. Accurate analysis of questions thus requires a large corpus of questions and associated short answers to develop high performance components for part of speech tagging, parsing and question typing.

Some recent work on question typing (associating questions with particular semantic classes of answers) has explored "found" corpora. For example, (Mann2001) used two corpora of trivia questions with short answers.[9] Other researchers have mined frequently asked questions (FAQs) as sources of question-answer sets. These are particularly useful in developing question answering systems in specific domains, to support on-line help systems or to partially automate help desks. In addition, there are an increasing number of question-answer web sites, for example, sites that provide tests for language learners, or news providers that host quizzes on current events. Additional work in mining or capturing such on-line collections would accelerate progress in question answering.

One obvious source of question-answer pairs is multiple choice questions. These are widely used in standardised tests, such as reading comprehension tests and Testing of English as a Foreign Language (TOEFL) tests[10]. However, multiple choice questions are not as natural as short answer questions, since they are primarily designed for ease of grading. Also, because standardised tests are expensive resources to create, it can be difficult to obtain corpora of such materials for use in research or evaluation.

5.2 Questions

The above discussion on resources assumes that any collection of question-answer pairs would be of interest; however, some question types are much more tractable than others. Research to date has focused mostly on the easier kinds of questions. In the first two TREC evaluations, for example, questions were limited to simple factual questions that had answers in the associated document collections. As a result, the best strategy for the evaluation was always to produce a ranked list of proposed correct answers, since no credit was given to the system for "knowing" that it was not certain of the answer.

[9] The specific web sites were www.triviaspot.com and www.phishy.net/trivia.
[10] see www.toefl.org for sample materials

The next evaluation (TREC-10, November 2001) will increase question complexity in two dimensions: allowing questions that have no answers, and allowing questions with "list" answers.

This additional complexity will require changes in answer evaluation and in system construction. To handle questions that do not have answers in the underlying collection (that is, where "NO ANSWER FOUND" is a correct answer), systems will need to measure their certainty about an answer.

Questions requiring a list as an answer (e.g., *list the countries bordering Afghanistan*) may need to cull the answers from multiple sentences or clauses in a single document or may need to synthesize the answer from multiple documents. Both of these changes require extending the simple model beyond merely finding a sentence or region of a document that best answers the question.

5.3 Answer Evaluation

The first problem in evaluation is to decide on the criteria for judging an answer. The following list captures some possible criteria for answer evaluation; see Breck et al. (2000) for a discussion of these criteria:

- Relevance: the answer should be a response to the question.
- Correctness: the answer should be factually correct.
- Conciseness: the answer should not contain extraneous or irrelevant information.
- Completeness: the answer should be complete – that is, a partial answer should not get full credit.
- Coherence: an answer should be coherent, so that the questioner can read it easily.
- Justification: the answer should be supplied with sufficient context to allow a reader to determine why this was chosen as an answer to the question.

So far, evaluations have focused primarily on relevance, although the TREC question answering evaluation now requires that the answer be justified within the document, and the byte limitation on answers goes a first step towards addressing the conciseness criterion. In some cases, optimizing along one criterion may reduce "goodness" along another dimension – for example, answer justification may reduce answer conciseness. Therefore, the criteria of correctness must be related to the intended use, the intended users, and the interface.

Once there is agreement on criteria for what constitutes a good answer, there need to be repeatable evaluation procedures. The Voorhees paper describes the TREC process which uses human assessors to read and evaluate each answer. Experiments during TREC-8 determined that consistency among the human assessors was good enough to preserve the relative ranking of systems. As a result, it was possible to have answers graded by only a single evaluator. This has significantly decreased cost, but because a human is required, this method does not support systematic iterative testing for hill climbing or machine learning.

It is always useful to provide human performance benchmarks for an evaluation task. In this regard, reading comprehension tests are ideal, since they are designed to evaluate people, e.g., children in school, or adult learners of a second language. However, these tests also require human assessors for evaluation, unless multiple choice tests are used.

5.4 Automated Evaluation Techniques

There is ongoing research in automated evaluation methods. For grading short answers, it is possible to automate comparison of system (or student) answers to answer keys created by a human expert (Breck et al.2000; Hirschman et al.2000). Such comparisons, while not as accurate as those done by human assessors as in TREC, still provide reasonably good agreement (93-95%) with human assessors – good enough for iterative training and machine learning.

Automated answer and essay grading is a topic of great significance to the educational testing community. Multiple choice tests are still widely used, despite agreement that short answer tests and essays are better tests. This is because open ended tests are felt to be too laborious or too subjective for use in large scale standardized testing. However, recent work in automated essay grading (Kukich2000; Laudauer and Laham2000) has demonstrated the feasibility of automated evaluation for essay tests, sometimes in conjunction with a single human assessor. These results provide the hope that automated answer grading systems could eventually approximate human graders. If we were able to construct systems that could evaluate or grade answers with results consistent with human performance, new possibilities open up: such systems could complement teachers in the class room by grading student exercises or allowing students to do self paced learning. And if a system could both answer questions accurately and evaluate the correctness of answers, it could even teach people, or at least provide a "learning companion" (Goodman et al.1998).

6 Future Directions

Despite 40 years of activity, we are just beginning to explore question answering as a research area. The attraction of the question answering challenge is that it is both tractable (witness the impressive performance evaluation results from the TREC question answering evaluation) and highly relevant: even the limited solutions developed to date provide significant value added over coarse-grained document retrieval. It is also stimulating cross-fertilisation of ideas between researchers in natural language processing, information retrieval and artificial intelligence.

Recent research in this area has been focused primarily on the TREC question answering evaluation, although to a lesser extent on other tasks, such as reading comprehension. Open "common" evaluations, such as TREC, are enormous drivers of progress. They bring a rigorous empirical approach to research, providing a challenge task, experimentation, empirical validation and comparison of results, and replication.

However, formal evaluations are always an abstraction of the real problems. They are primarily designed for the ease and replicability of evaluation. The current evaluations are only the first stages of an ambitious plan to evaluate many dimensions of question answering.[11] It is important to review the larger agenda of question answering, which goes far beyond our current ability to build or evaluate such systems. As a research agenda, question answering poses long-term research challenges in many critical areas of natural language processing:

- Applications: section 3 provided an overview of the use of question answering to access structured information (in databases) as well as free text. Other applications include automated help, Web content access (by both speech and text), front-ends to knowledge sources such as on-line encyclopedias or to bibliographic resources (e.g., to MEDLINE for biomedical literature). Important applications also exist in the educational world for language teaching and for companion learning systems. Automated evaluation techniques will support short answer grading, as well as automated essay grading. Future question answering systems should be able to access content in multiple languages and across multiple media as well.
- Users: Current question answering systems provide answers to isolated factual questions. Real users want real-time interactive question and answer capabilities, with coherent succinct answers presented in context for easy inspection. Power users will need systems that have constantly updated user models, so that the system presents only novel information. Other sets of users will want digests or background summaries of information, or information organized by time or by location. Users new to a domain – learners – may need incorrect factual or conceptual presuppositions of their questions identified and corrected; and they need responses tailored to their level of comprehension. Once question answering front-ends become a part of the user's environment, users will want support for collaborative question answering, to allow teams of people to research questions, share information and integrate partial answers. This will require progress in areas such as real-time architectures for question answering, user modeling, and collaborative work environments.
- Question types: Question types will move from the factual to more complex forms of question, including lists, summarization of contradictory information, and explanations, including answers to *how* or *why* questions, and eventually, *what if* questions. Systems will have to recognize paraphrases of the same underlying question, including cross-language question answering, which would allow the user to ask a question in their native language, access information from documents in multiple languages and receive an answer in their native language. This will require progress in cross language retrieval and machine translation.

[11] See the question answering Roadmap document (Burger et al.2001) for incremental expansion of the question answering task. There is now also a significant research effort (AQUAINT) starting in the US focused on question answering; see www.ic-arda.org/solicitations/AQUAINT/

- Answer types: One of the greatest challenges may be in presenting appropriate answers. Answers must be correct, succinct, coherent and justified (either within the answer or as a pointer to the source(s) of information). Systems will need to handle cases when multiple answers are found, when no answer is found, or when contradictory answers are found. Systems will have to go beyond extraction of snippets of text to provide answer synthesis across sentences and across documents. Providing appropriate coherent answers will be a major research area and will depend heavily on progress in text summarization. Question answering systems could also go beyond text-based sources, to include other media: spoken language, imagery, data from structured sources, including databases and knowledge bases. And the ideal question answering system would be able to retrieve and merge these answers into the appropriate (multimedia) form for the end user.
- Evaluation: With the introduction of each new set of features, the evaluation paradigm will have to be adapted to evaluate the enhanced capabilities of the systems. There are two particularly important challenges here. First, better automated evaluation is needed to support machine learning and statistical methods. If successful, automated evaluation methods will enable new applications in the education and training fields. Second, a user-centered evaluation method needs to be developed, so that user concerns (speed of response, answer display, support for interactivity and collaboration, usability of answer) can be evaluated. Without such user-centered evaluations, an important dimension of research will be neglected.
- Presentation: Ultimately, question answering is about providing information to users. The success in meeting users' needs will provide the market "pull" that drives this area forward. It is important to understand what users need and develop user-centered evaluations to drive this work. Further research is needed that will draw heavily on work in interactive retrieval, answer presentation and summarization, conversational interfaces, and human-computer interaction in general.

From this list, we see that the long-term view of question answering intersects with many areas of natural language processing, knowledge representation, human-computer interaction, multimedia processing, collaborative systems, and intelligent tutoring systems. This issue represents a snapshot of this area at an early stage in this ambitious research agenda: "the view from here".

Acknowledgements

The authors would like to thank the following for their invaluable help in reviewing papers for this special issue: Giuseppe Attardi, Eric Breck, Ted Briscoe, John Carroll, Eugene Charniak, Robert Dale, Brigitte Grau, Sanda Harabagiu, Ed Hovy, Christian Jacquemin, Guy Lapalme, Marc Light, Michael Littman, Paul Martin, Gideon Mann, Hwee Tou Ng, John Prager, Ellen Riloff, Tomas Strzalkowski, John Tait, Yorick Wilks.

References

N. D. Belnap and T. B. Steel. 1976. *The Logic of Questions and Answers.* Yale University Press, New Haven and London.

D. Bobrow, Kaplan R, M. Kay, D. Norman, H. Thompson, and T. Winograd. 1977. GUS, a frame driven dialog system. *Artificial Intelligence*, 8:155–173.

E. Breck, J. Burger, L. Ferro, L. Hirschman, D. House, M. Light, and I. Mani. 2000. How to evaluate your question answering system every day ... and still get real work done. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC-2000)*, pages 1495–1500.

J. Burger, C. Cardie, V. Chaudhri, R. Gaizauskas, S. Harabagiu, D. Israel, C. Jacquemin, C-Y Lin, S. Maiorano, G. Miller, D. Moldovan, B. Ogden, J. Prager, E. Riloff, A. Singhal, R. Shrihari, T. Strzalkowski, E. Voorhees, and R. Weishedel. 2001. Issues, tasks and program structures to roadmap research in question & answering (Q&A). Available at: http://www-nlpir.nist.gov/projects/duc/roadmapping.html.

C. Clarke, G. Cormack, D. Kisman, and T. Lynam. 2001. Question answering by passage selection (multitext experiments for trec-9. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9).* NIST Special Publication 500-XXX. Available at: http://trec.nist.gov/pubs.html.

A. Copestake and K. Sparck Jones. 1990. Natural language interfaces to databases. *The Knowledge Engineering Review*, 5(4):225–249.

O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, and C. Jacquemin. 2001. Terminological variants for document selection and question/answer matching. In *Proceedings of the Association for Computational Linguistics Workshop on Open-Domain Question Answering*, pages 46–53.

R. Gaizauskas and Y. Wilks. 1998. Information Extraction: Beyond Document Retrieval. *Journal of Documentation*, 54(1):70–105.

B. Goodman, A. Soller, F. Linton, and R. Gaimari. 1998. Encouraging student reflection and articulation using a learning companion. *International Journal of Artificial Intelligence in Education*, 9:237–255.

B. F. Green, A. K. Wolf, C. Chomsky, and K. Laughery. 1961. BASEBALL: An automatic question answerer. In *Proceedings of the Western Joint Computer Conference* **19**, pages 219–224. Reprinted in (Grosz et al.1986), pages 545-549.

C. Green. 1969. Theorem proving by resolution as a basis for question-answering systems. *Machine Intelligence*, 4:183–205.

B.J. Grosz, K. Sparck Jones, and B.L. Webber, editors. 1986. *Readings in Natural Language Processing.* Morgan Kaufmann, Los Altos, CA.

C. Hamblin. 1967. Questions. In P. Edwards, editor, *The Encyclopedia of Philosophy*, pages 49–53. Macmillan, New York.

S. Harabagiu, D. Moldovan, M. Paşca, R. Mihalcea, M. Surdeanu, R. Bunescu, R. Gîrji, V. Rus, and P. Morărescu. 2001. FALCON: Boosting knowledge for answer engines. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9).* NIST Special Publication 500-XXX. Available at: http://trec.nist.gov/pubs.html.

D. Harrah. 1984. The logic of questions. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, Vol. II*, pages 715–764. D. Reidel.

U. Hermjacob. 2001. Parsing and question classification for question answering. In *Proceedings of the Association for Computational Linguistics Workshop on Open-Domain Question Answering*, pages 17–22.

L. Hirschman, M. Light, E. Breck, and J. Burger. 1999. Deep Read: A reading comprehension system. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 325–332.

L. Hirschman, E. Breck, M. Light, J. Burger, and L. Ferro. 2000. Automated grading of short answer test. *IEEE Intelligent Systems*, 15(5):31–34.

E. Hovy, L. Gerber, U. Hermjacob, M. Junk, and C-Y. Lin. 2001. Question answering in Webclopedia. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9)*. NIST Special Publication 500-XXX. Available at: http://trec.nist.gov/pubs.html.

A. Ittycheriah, M. Franz, W-J. Zhu, and A. Ratnaparkhi. 2001. IBM's statistical question answering system. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9)*. NIST Special Publication 500-XXX. Available at: http://trec.nist.gov/pubs.html.

B. Katz. 1997. From sentence processing to information access on the world wide web. In *Proceedings of the AAAI Spring Symposium on Natural Language Processing for the World Wide Web, Stanford University, Stanford CA*. Available at: http://www.ai.mit.edu/people/boris/webaccess/.

W. Kintsch. 1998. *Comprehension : A Paradigm for Cognition*. Cambridge University Press, Cambridge.

K. Kukich. 2000. Beyond automated essay scoring. *IEEE Intelligent Systems*, 15(5):22–27.

J. Kupiec. 1993. Murax: A robust linguistic approach for question answering using an on-line encyclopedia. In *Proceedings of ACM-SIGIR '93*, pages 181–190, Pittsburgh, PA.

T. Laudauer and D. Laham. 2000. The intelligent essay assessor. *IEEE Intelligent Systems*, 15(5):27–31.

W. Lehnert. 1977. A conceptual theory of question answering. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pages 158–164. Reprinted in (Grosz et al.1986), pages 651-657.

M. Light, E. Brill, E. Charniak, M. Harper, E. Riloff, and E. Voorhees, editors. 2000. *Proceedings of the Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, Seattle. Association for Computational Linguistics.

I. Mani, T. Firmin, D. House, G. Klein, B. Sundheim, and L. Hirschman. To appear. The TIPSTER SUMMAC text summarization evaluation. *Journal of Natural Language Engineering*.

G. S. Mann. 2001. A statistical method for short answer extraction. In *Proceedings of the Association for Computational Linguistics Workshop on Open-Domain Question Answering*, pages 23–30.

D. Milward and J. Thomas. 2000. From information retrieval to information extraction. In *Proceedings of the ACL Workshop on Recent Advances in Natural Language Processing and Information Retrieval*. Available at: http://www.cam.sri.com/html/highlight.html.

D. Moldovan, S. Harabagiu, M. Paşca, R. Mihalcea, R. Goodrum, R. Gîrji, and V. Rus. 2000. LASSO: A tool for surfing the answer net. In *Proceedings of the Eighth Text Retrieval Conference (TREC-8)*. NIST Special Publication 500-246. Available at: http://trec.nist.gov/pubs.html.

D. Molla Aliod, J. Berri, and M. Hess. 1998. A real world implementation of answer extraction. In *Proceedings of the 9th International Conference on Database and Expert Systems Applications Workshop "Natural Language and Information Systems" (NLIS'98)*, pages 143–148.

J. Prager. 2001. One search engine or two for question answering. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9)*. NIST Special Publication 500-XXX. Available at: http://trec.nist.gov/pubs.html.

M. L. Prior and A. N. Prior. 1955. Erotetic logic. *The Philosophical Review*, 64(1):43–59.

L. Åqvist. 1975. *A New Approach to the Logical Theory of Interrogatives*. TBL Verlag Gunter Narr, Tübingen.

R.C. Schank and R.P. Abelson. 1977. *Scripts, Plans, Goals, and Understanding*. Lawrence Erlbaum Associates, Hillsdale, N.J.

S. Scott and R. Gaizauskas. 2001. University of Sheffield TREC-9 Q & A System. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9)*. NIST Special Publication 500-XXX. Available at: http://trec.nist.gov/pubs.html.

R. F. Simmons. 1965. Answering english questions by computer: A survey. *Communications of the ACM*, 8(1):53–70.

K. Sparck Jones and P. Willett, editors. 1997. *Readings in Information Retrieval*. Morgan Kaufmann, San Francisco.

R. Srihari and W. Li. 2000. Information extraction supported question answering. In *Proceedings of the Eighth Text Retrieval Conference (TREC-8)*. NIST Special Publication 500-246. Available at: http://trec.nist.gov/pubs.html.

A. Turing. 1950. Computing machinery and intelligence. *Mind*, 59(236):433–460.

E. M. Voorhees and D. K. Harman, editors. 2000. *Proceedings of the Eighth Text Retrieval Conference (TREC-8)*. NIST Special Publication 500-246. Available at: http://trec.nist.gov/pubs.html.

B. L. Webber. 1986. Questions, answers and responses: Interacting with knowledge-base systems. In M. L. Brodie and J. Mylopoulos, editors, *On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies*, pages 365–402. Springer-Verlag.

T. Winograd. 1972. *Understanding Natural Language*. Academic Press, New York.

W. Woods. 1973. Progress in natural language understanding – an application to lunar geology. In *AFIPS Conference Proceedings*, volume 42, pages 441–450.

R. Zajac. 2001. Towards ontological question answering. In *Proceedings of the Association for Computational Linguistics Workshop on Open-Domain Question Answering*, pages 31–37.

V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. J. Hazen, and L. Heatherington. 2000. JUPITER: A telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing*, 8(1):100–112.