Question Answering from the Web Using Knowledge Annotation and Knowledge Mining Techniques

Jimmy Lin and Boris Katz
MIT Computer Science and Artificial Intelligence Laboratory
200 Technology Square
Cambridge, Massachusetts, USA
{jimmylin,boris}@ai.mit.edu

ABSTRACT

We present a strategy for answering fact-based natural language questions that is guided by a characterization of real-world user queries. Our approach, implemented in a system called Aranea, extracts answers from the Web using two different techniques: *knowledge annotation* and *knowledge mining*. Knowledge annotation is an approach to answering large classes of frequently occurring questions by utilizing semistructured and structured Web sources. Knowledge mining is a statistical approach that leverages massive amounts of Web data to overcome many natural language processing challenges. We have integrated these two different paradigms into a question answering system capable of providing users with concise answers that directly address their information needs.

Categories and Subject Descriptors

H.3.4 [**Information Systems**]: Information Storage and Retrieval, Systems and Software

General Terms

Design, Performance

Keywords

semistructured data, data-redundancy

1. INTRODUCTION

The vast amounts of information available on the World Wide Web makes it an attractive resource for answering a variety of questions that users may have. However, the effectiveness of such a "knowledge repository" is limited by practical means of information access. The sheer size of the Web threatens to overwhelm both causal users and information professionals alike; Web search engines frequently return hundreds of thousands of documents in response to a query. To satisfy an information need, users are often forced to engage in the labor-intensive task of manually perusing "potentially relevant" documents returned by keyword-based search engines.

Question answering has recently emerged as a technology that promises to provide more intuitive methods of information access. In contrast to the traditional information retrieval model of formulating queries and browsing results, a question answering system simply accepts user information requests phrased in everyday language and responds with a concise answer. Currently, question answering research has focused on fact-based questions like "When did Montana become a state?" or "How far is it from the pitcher's mound to home plate?" Most of these "factoid" questions can be answered with simple entities such as dates, locations, people, organizations, measures, etc.

This paper describes Aranea, an open-domain question answering system that takes advantage of Web data to answer factoid questions. Our system embraces two different views of the World Wide Web: as a heterogenous collection of unorganized documents and as a source of carefully crafted and organized knowledge about specific topics. To take advantage of these different facets of the Web, our system integrates two different paradigms of question answering: *knowledge annotation* using semistructured database techniques and *knowledge mining* using redundancy-based statistical techniques.

Aranea's approach to question answering is primarily motivated by an observation about the empirical distribution of user queries, which turns out to quantitatively obey Zipf's Law—a small fraction of question types accounts for a significant portion of all question instances. Many questions ask for the same type of information, differing only in the specific object questioned, e.g., "What is the population of the United States?", "What is the population of Mexico?", "What is the population of Canada?", etc. Not only do such questions appear frequently, but they can also be naturally grouped together into a single type or class, i.e., "What is the population of x?" where x is a variable that can stand in for any country. Figure 1 presents an analysis of testsets from the TREC-9 and TREC-10 question answering evaluations, where unique question types are plotted against cumulative distribution of all questions. We can see, for example, that twenty question types account for over twenty percent of questions from the TREC-9 and forty percent of the questions from the TREC-10 testsets. Analysis of logs from the START question answering system [15], which has answered

CIKM'03, November 3–8, 2003, New Orleans, Louisiana, USA.

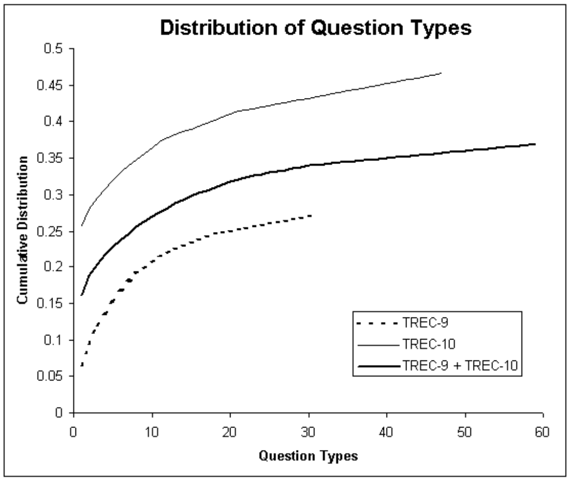

Figure 1: Cumulative distribution of typical user questions: number of question types (or classes) plotted against the cumulative percentage of the testset that those question types account for.

millions of questions over the last decade, and commercial search engine logs [25] leads to the same conclusions. Large classes of commonly-occurring questions translate naturally into database queries and are handled by Aranea using a technique we call *knowledge annotation*, which allows our system to access semistructured and heterogenous data as if it were a uniform database.

As with all Zipf curves, there is a broad tail where individual instances are either unique or account for an insignificant fraction of total questions. In addition to asking large classes of commonly-occurring questions, users also pose a significant number of unique questions that cannot be easily classified into common categories or grouped by simple patterns. To answer these questions, Aranea employs what we call redundancy-based *knowledge mining* techniques. Knowledge mining leverages the massive amounts of information available on the Web to overcome many thorny problems associated with natural language processing.

This paper describes the workings of our Aranea question answering system and presents the effectiveness of our approach, as evaluated at the 2002 Text Retrieval Conference (TREC-2002) [28]. The question answering track at TREC, which began in 1999, is an annual event designed to evaluate question answering systems against a common testset using a shared methodology, with the goal of encouraging discourse within the research community.

2. OVERALL FRAMEWORK

The overall architecture of Aranea is shown in Figure 2. User questions are sent to two separate components, one that employs knowledge annotation (described in Section 3) and one that utilizes knowledge mining (described in Section 4). Both components consult the World Wide Web to generate candidate answers, which are then piped through a knowledge boosting module (Section 5) that checks the candidate answers against a number of heuristics to ensure their validity.

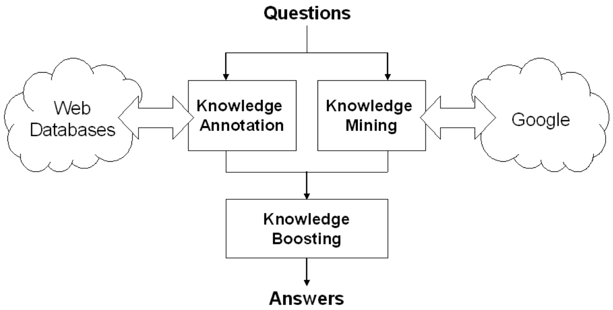

Figure 2: Overall architecture of Aranea.

3. KNOWLEDGE ANNOTATION

Although the Web consists largely of unorganized pages, pockets of structured and semistructured knowledge exist as valuable resources for question answering. For example, the CIA World Factbook provides political, geographic, and economic information about every country in the world; 50states.com contains numerous properties related to US states from state bird to land area; Biography.com has collected profiles of over twenty-five thousand famous people; the Internet Movie Database stores entries for hundreds of thousands of movies, including information about their cast, production staff, and dozens of other properties.

To effectively use these resources, a system must integrate them under a common interface. Drawing from database concepts, we have developed a schema-based technique called knowledge annotation that connects natural language queries with semistructured knowledge sources. This technique is a simplified implementation of the technologies pioneered by START [15] and Omnibase [16], two components of the world's first Web-based question answering system.

Since it came on-line in December, 1993, START[1] has engaged in exchanges with hundreds of thousands of users all over the world, supplying them with useful knowledge. However, because the system provides users with paragraph-sized answers that often contain multimedia fragments such as pictures and audio clips, it is not suitable for a TREC-style evaluation. In general, we believe that paragraph-sized chunks form the most suitable unit of response to a user question, because complementing the short answer with additional contextual information may help with interpretation and analysis [24]. However, because the TREC QA track accepts only *exact answers*, we found it inappropriate to directly evaluate START and Omnibase. Instead, we have channeled our experiences into Aranea, a broader coverage, but less linguistically-sophisticated question answering system designed specifically to return TREC-style answers.

3.1 Database Access Schemata

The heart of Aranea's knowledge annotation component is a collection of database access schemata. Each schema is composed of two connected parts: the question signature and the database query. A question signature is a collection of regular expressions that match a specific class of user questions, e.g., requests for birth dates of people.[2] These

[1] `http://www.ai.mit.edu/projects/infolab/`

[2] These question signatures are a simplified version of the natural language annotations used by START [15], which

annotations are paired with unfilled database queries that are dynamically instantiated with bindings extracted from the question signature. Consider a typical schema:

> When was x born?
> What is the birth date of x?
> . . .
> → (biography.com x birthdate)

In this example, questions that ask for birth dates are translated into an *object–property–value* database query [16]. These queries specify the data source where the answer could be found (biography.com), the *object* in question (x), and the *property* sought after (birthdate). The *value* of the object's property typically answers the user's question. In practice, we have discovered that this data model is expressive enough to capture a significant fraction of semistructured Web resources as well as user questions.

The knowledge annotation component of Aranea operates by matching user questions against stored schemata and executing database queries generated as a result. The execution of these queries varies with the data source: Some sources are stored locally and translate into a simple file lookup. Other sources are stored on remote Web sites behind CGI scripts; executing queries on these sources requires dynamically reconstructing an HTTP request and properly parsing the resulting HTML document. Because each resource is structured differently, wrappers must be manually crafted for each individual data source.

The Aranea system deployed for the 2002 TREC evaluation included twenty-eight schemata that access seven different data sources. Here are two examples:

- **Biography.com** provides information about the birth dates, death dates, etc., of various well-known people. Answering questions about such properties involves dynamically retrieving pages from biography.com (via CGI) and performing simple pattern matching on the HTML to extract exact dates.

- **CIA World Factbook** provides various useful facts about countries, e.g., population, area, capital, etc. This information was downloaded and structured into a locally-stored tab-delimited file. Questions about various properties of world countries are translated into simple file lookups.

Despite the manual labor involved in "wrapping" data sources, knowledge annotation nevertheless remains an effective question answering strategy. Because users often ask similar questions, a few well-chosen knowledge sources suffice to answer a significant fraction of questions. For example, we have verified that ten Web sources can provide answers to 27% of TREC-9 and 47% of TREC-2001 QA track questions [23]. In addition, the importance of specific knowledge sources has been noticed by other researchers as well [12, 8]. By letting the distribution of user questions guide our wrapper development, we can achieve good performance with modest amounts of knowledge engineering.

3.2 Related Work

The knowledge annotation strategy for question answering was first introduced and implemented in Start [15] and Omnibase [16]. With the adaptation of our techniques in the Aranea system, we have had, for the first time, an opportunity to evaluate the technology in a formal setting at the TREC 2002 question answering competition.

The idea of applying database techniques to the World Wide Web is not new (cf. [10]). Many existing systems, e.g., Araneus [1], Ariadne [20], Tsimmis [11], just to name a few, have attempted to integrate heterogeneous Web sources under a common interface. Unfortunately, queries to such systems must be formulated in SQL, Datalog, or some similar formal language, which render them inaccessible to the average user. The unique contribution of our approach is the integration of database and natural language techniques.

The viability of our annotation-based question answering technique has also been demonstrated commercially. For example, Ask Jeeves, currently the Web's second most popular search engine, licenses certain technologies [18, 19] pioneered by the Start system.

4. KNOWLEDGE MINING

The knowledge mining approach to question answering is a data-driven strategy that capitalizes on the enormous amounts of text freely available on the World Wide Web.

One of the biggest challenges in question answering is relating the formulation of a question to different formulations of its answers. Consider a question like "When did Alaska become a state?": if a document plainly stated that "Alaska became a state on January 3, 1959", the answer would be relatively easy to extract. Unfortunately, the expressiveness of natural language allows the same meaning to be expressed in a variety of different ways; more likely, the answer to the above question would be stated as "Alaska was admitted to the Union on January 3, 1959." Since this answer shares few keywords in common with the question, a system would require sophisticated reasoning, e.g., the ability to recognize paraphrases, in order to relate the answer to the question. The knowledge mining approach to question answering attempts to overcome this difficulty by leveraging the massive size of the Web.

The most important implication of the Web's size is data redundancy—each item of information has potentially been stated in many ways, in many different documents. A question answering system can capitalize on this redundancy in two ways: as a surrogate for sophisticated natural language techniques and as a method for overcoming poor document quality. Consider the question "When did Wilt Chamberlain score 100 points?" Here are two possible answers:

> (1) Wilt Chamberlain scored 100 points on March 2, 1962 against the New Yorks Knicks.
>
> (2) On December 8, 1961, Wilt Chamberlain scored 78 points in a triple overtime game. It was a new NBA record, but Warriors coach Frank McGuire didn't expect it to last long, saying, "He'll get 100 points someday." McGuire's prediction came true just a few months later in a game against the New York Knicks on March 2.

Obviously, the answer could be more easily extracted from sentence (1) than from passage (2). In general, the task of

are parsed into and stored as ternary expressions. Because matching occurs at the level of the parsed structures, powerful linguistic machinery can be employed to handle different linguistic phenomena, e.g., synonymy, hyper/hyponymy, alternations, etc. In Aranea, we have attempted to approximate natural language processing techniques with regular expressions.

answering a question is not very difficult if the document collection contains the answer stated as a simple reformulation of the question. In these cases, simple keyword-based techniques coupled with named-entity detection technology suffice to identify the answer. However, without the luxury of massive amounts of data, a question answering system may be forced to extract answers from passages in which they are not obviously stated, e.g., passage (2). In these cases, sophisticated natural language processing may be required, e.g., recognizing syntactic alternations, resolving anaphora, making commonsense inferences, performing relative date calculations, etc.

The Web is so big that simple pattern matching techniques can often obviate the need to understand both the structure and meaning of language. With enough data, there is a good chance that an answer will appear as a simple reformulation of the question. In such cases, the answer could be extracted by searching directly for an anticipated answer form, e.g., in the above example, by searching for the string "Wilt Chamberlain scored 100 points on" and extracting words occurring to the right. Naturally, this simple technique depends crucially on the corpus having an answer formulated in a specific way—the larger the text collection is, the greater the probability that simple pattern matching techniques will yield the correct answer. Data redundancy enables a simple trick to overcome many troublesome issues in natural language processing.

The effect of data redundancy has been quantified by other researchers as well. Breck *et al.* [3] noticed a correlation between the number of times an answer appeared in the TREC corpus and the average performance of TREC systems on that particular question. Similarly, Clarke *et al.* [7] noticed an upward trend in performance as a question answering system was given a larger corpus from which to extract answers. These results verify our intuition: the more times an answer appears (in different formulations), the easier it is to find it.

In a Web environment, data redundancy also serves as a guard against erroneous information. Because the overall quality of documents on the Web is lower than typical closed corpora (e.g., collections of newspaper articles), any single instance of an answer is inherently untrustworthy. However, because a fact is usually stated multiple times in multiple documents, a question answering system could utilize the distribution of answers across multiple sources to gauge its reliability.

The tremendous amounts of information on the World Wide Web would be useless without an effective method of data access. Providing the basic infrastructure for indexing and retrieving text at such scales is a tremendous engineering task. Fortunately, such services already exist, in the form of search engines. Using existing search engines as information retrieval backends, we can focus our efforts on answer extraction.

4.1 Knowledge Mining Modules

The data flow in the knowledge mining component of Aranea is shown in Figure 3. In the following sections, we describe each module in detail.

4.1.1 Formulate Requests

The first step in answering factoid questions is to translate them into queries, or requests. These requests specify the context in which answers are likely to be found, and are analogous to queries posed to traditional information retrieval systems. However, because Aranea relies on Web search engines to fulfill these requests, fine-grained control over the query and result set is difficult; Aranea instead relies on *quantity* to make up for lack of *quality*.

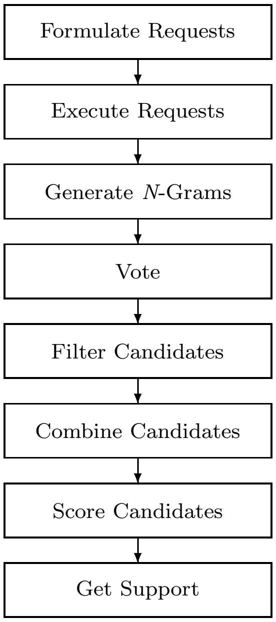

Figure 3: Data flow in the knowledge mining component of Aranea.

Two types of queries are generated by this module: *exact* queries and *inexact* queries. Queries of both types are concurrently generated, but usually given different scores. An inexact query indicates that the answer is likely to be found within the vicinity of a set of keywords. They are composed by treating the natural language question as a bag of words. In contrast, an exact query details the specific location of a potential answer, e.g., the answer to "When did the Mesozoic period end?" is likely to appear within ten words and fifty bytes to the right of the exact phrase "the Mesozoic period ended". Exact queries in Aranea are generated by approximately a dozen pattern matching rules based on query terms and their part-of-speech tags; morpho-lexical pattern matches trigger the creation of reformulated exact queries. As an example, the previous query was generated by the rule "*wh-word* did . . . *verb* → . . . *verb*+ed". An internal lexicon ensures that the generated verb remains properly conjugated.

As a complete example, the requests generated in response to the question "When did the Mesozoic period end?" are shown in Figure 4. Aranea generates two inexact and one exact request; each query is also assigned a basic score, which helps establish the relative importance of the queries.

4.1.2 Execute Requests

The request execution module is responsible for retrieving textual snippets that honor the constraints set forth by each request. Currently, Google is used to mine text from the Web. In the case of inexact requests, the entire summary

Query: When did the Mesozoic period end
  Type: inexact
  Score: 1
  Number of snippets to mine: 100
Query: the Mesozoic period ended
  Type: inexact
  Score: 1
  Number of snippets to mine: 100
Query: the Mesozoic period ended ?x
  Type: exact
  Score: 2
  Number of snippets to mine: 100
  Maximum length for `?x`: 50
  Maximum word count for `?x`: 5

Figure 4: Typical requests generated by Aranea.

provided by Google is extracted for further processing. For exact queries, the request execution module performs additional pattern matching to ensure that the correct positional constraints are satisfied.

4.1.3 Generate N-Grams

This module exhaustively generates all possible unigrams, bigrams, trigrams, and tetragrams from the text fragments generated by the request execution module. These n-grams, which are given initial scores equal to the weight of the request from which they derive, serve as candidate answers.

4.1.4 Vote

The voting module collates the n-grams generated by the previous module. The new score of each answer candidate is equal to the sum of the scores of all occurrences of that particular n-gram. This module has the effect of promoting text fragments that occur frequently (in the context of query terms), and are hence more likely to answer the user question.

This process of voting is meant to counteract the low average quality of individual documents. Many Web documents are poorly written, barely edited, or simply contain incorrect information. Although text extracted from a single document cannot be trusted as the correct answer, multiple occurrences of the same answer in different documents lends credibility to the proposed answer.[3]

4.1.5 Filter Candidates

In this stage of processing, a coarse-grained filter is applied to the answer candidates:

- Candidates that begin or end with stopwords are discarded.
- Candidates that contain words found in the original user question are discarded. The only exception to this rule is question focus words, e.g., a question beginning with "How many meters…" can be answered by an expression containing the word "meters".

This stage also encodes a few heuristics that can potentially decrease the number of answer candidates. For example, the answer to "how far", "how fast", "how tall", etc., questions must invariably contain a numeric component (either numeric digits or numerals); thus, we can safely discard all answer candidates that do not fit this criteria. The heuristics employed by this module tend to filter with high confidence, erring on the side of being too lenient. False positive results can always be sorted out by later modules, but the system will not be able to recover from false negatives.

In addition, a set of fixed-list filters is applied to questions whose answer types are closed-class items. For example, a question like "What language do most people speak in Brazil?" must be answered with a language; thus, we can safely throw out any answer candidate that isn't a language. For a variety of question types, e.g., "What sport…", "What nationality…", it is relatively straightforward to enumerate all acceptable answer candidates. In these cases, fixed lists can be used as high-precision filters to throw out irrelevant candidates. We have implemented roughly a dozen of such filters in Aranea.

4.1.6 Combine Candidates

In this module, shorter answers are used as evidence to boost the score of longer answers. If a portion of a candidate answer appears itself as a candidate answer, then the score of the shorter answer is added to the score of the longer answer. For example, if "de Soto" appears on the list of candidate answers along with "Hernando de Soto", the score of the shorter candidate would be added to the score of the longer one. This module counteracts the tendency of the n-gram generation and voting modules to favor shorter answers.

4.1.7 Score Candidates

The score of each answer candidate is multiplied by the following factor:

$$\frac{1}{|A|}\sum_{w \in A} \log(\frac{N}{w_c})$$

A is the set of keywords in the candidate answer; N is the total number of words in our corpus; w_c is the number of occurrences of word w in the corpus. This scoring balances the effect of individual keywords having different (unconditioned) priors. Since the exact distribution of unigrams on the Web can not be easily obtained in a reliable manner, Aranea uses statistics from our corpus as a surrogate. We used the official corpus of the TREC question answering track—the AQUAINT corpus—which is comprised of roughly one million articles from the New York Times, the Associated Press, and the Xinhua News Service.

4.1.8 Get Support

This module performs a final sanity check on the candidate answers. It verifies that final candidate answers actually appear in the original text snippets mined from the Web. Occasionally, the various modules within the knowledge mining component of the system will assemble a nonsensical answer; this module discards such answers.

4.2 Related Work

Many other works [5, 6, 7, 22, 4] are similar in spirit to the knowledge mining paradigm employed by Aranea. In general, we view the knowledge mining component of Aranea as the next generation of redundancy-based techniques for the World Wide Web. MULDER [22], one of the earliest

[3] Unfortunately, this technique equates the *most popular* answer with the *correct* answer, which occasionally results in very comical responses.

question answering systems to take advantage of commercial search engines, attempted to perform sophisticated linguistic analysis on both questions and potential answer candidates. As a result, it did not take advantage of data redundancy. Furthermore, the system was not formally evaluated on standardized testsets, and therefore its performance cannot be compared against other systems in a meaningful way. Shapqa [6], another system that attempted to apply linguistically-sophisticated techniques to answer extraction, performed worse than the average system at TREC-2001. In contrast, the AskMSR [5],[4] one of the top performers at TREC-2001, embraced data-redundancy and applied extremely simple word-counting techniques on Web data. However, it was an ad-hoc agglomeration of two separate systems whose output were then stitched together by yet another "combiner" system. As such, the performance contribution of various components was difficult to determine. In comparison, Aranea boasts a modular architecture that also serves as a testbed for a variety of knowledge mining techniques. MultiText [7], another question answering system that takes advantage of data redundancy, employs a different approach: instead of using the Web directly to answer questions, it treated the Web as an auxiliary corpus to validate candidate answers extracted from a primary, more authoritative, corpus.

With Aranea, we have taken advantage of previous experiences to refine the knowledge mining paradigm within a better engineered framework. Our system supports a modular architecture that allows specific functionality to be encoded into manageable components. This not only allows for faster development cycles, but facilitates glass-box testing to properly determine the effectiveness of various techniques.

5. ANSWER BOOSTING

Results from both the knowledge annotation and knowledge mining components of Aranea are subjected to a series of heuristic checks.

Within the answer boosting module of Aranea, a set of procedures is specifically dedicated to detecting and verifying geographic locations. We have gathered large lists of known geographic entities, e.g., countries, cities, U.S. states, Canadian provinces, etc. Using these lists, we were able to construct accurate recognizers for locative expressions. In response to location questions such as "Where is the Isle of Man?" or "Where is Toronto?", Aranea can identify answer candidates of the correct type and "boost" their scores. Unlike filtering (with fixed-lists), this process is heuristic-based, and can recognize a greater range of expressions, e.g., "in the Irish Sea" or "on the north shore of Lake Ontario". Using heuristics to "boost" potentially relevant answers is preferable to filtering out irrelevant answers in cases where enumeration of all acceptable responses is impractical.

Questions requiring dates as answers similarly receive special treatment. Since dates and temporal expressions have relatively fixed form, it is straightforward to detect such entities and promote them as potentially relevant answers. Knowledge of dates also helps Aranea extract exact answers. For example, a candidate answer to a "What year..." question often contains extra information such as the month and day; Aranea removes this extraneous information.

		# of q.	%
Knowledge Annotation	correct	30	6.0%
	inexact	2	0.4%
	wrong	10	2.0%
	total	42	8.4%
Knowledge Mining	correct	153	30.6%
	inexact	43	8.6%
	wrong	262	52.4%
	total	458	91.6%
Total	correct	183	36.6%
	inexact	45	9.0%
	wrong	272	54.4%
	total	500	100%

Table 1: Results from TREC-2002 question answering track.

		performance
Knowledge Annotation	correct	71.4%
	inexact	4.7%
	wrong	23.9%
Knowledge Mining	correct	33.4%
	inexact	9.4%
	wrong	57.2%

Table 2: Performance of individual components.

6. RESULTS

The Aranea system participated in the question answering track at TREC-2002 [28]. The annual track not only conducts formal evaluations of question answering systems, but also serves as a focal point for question answering research to facilitate the dissemination of results; see [29, 27, 28] for overviews of results from recent years. Notable changes in the 2002 evaluation is the *exact* answer requirement (explained below) and single-response requirement (i.e., each system is only allowed to return one answer for each question). A third new aspect of the evaluation, confidence-weighted scoring, is a relatively experimental procedure and not discussed here.

Results of the evaluation are shown in Table 1. In the TREC QA track, an answer was judged *correct* only if the system produced a document from the AQUAINT corpus that supported the answer. If the answer string was correct, but the supporting document did not confirm the answer, it would be judged as *unsupported*. Since Aranea extracts answers from the Web, the system subsequently needed to "project" the Web answers back onto the AQUAINT corpus to find a supporting document. We believe that this answer projection process is an artifact of the TREC evaluations, and argue that the process of answer projection is not directly part of the question answering task. Therefore, the results presented here are evaluations of the answer only; we have manually rescored *unsupported* judgments into either *exact* or *inexact*, disregarding the supporting document.

As an example of the difference between *exact* and *inexact* judgments, consider the question "What province in

[4] One of the authors of this paper was on the team that designed the AskMSR system.

Canada is Niagara Falls located in?": "southern Ontario" would be judged as *inexact* whereas "Ontario" would be judged as *exact*. Aranea's results show a relatively large number of inexact answers, which could be rectified by some superficial linguistic processing. For example, inexact answers often contain additional leading adjectives or common nouns, e.g., "western Montana" or "Baptist leader Roger Williams". These extraneous words could be easily stripped off with the help of a part-of-speech tagger and fixed lists of occupations, cardinal directions, etc.

Individual analysis of the knowledge annotation and knowledge mining components of Aranea is shown in Table 2. In general, the knowledge annotation component achieves much higher accuracy than the knowledge mining component. However, knowledge mining achieve broader coverage than knowledge annotation.

Here are some typical answers given by Aranea:

> **When is Gerald Ford's birthday?**
> July 14, 1913
> (extracted using knowledge annotation techniques from `biography.com`)
>
> **Who founded Taoism?**
> Lao Tzu
> (extracted using knowledge mining techniques)
>
> **What was the name of the first child of English parents to be born in America?**
> Virginia Dare
> (extracted using knowledge mining techniques)

Approximately 16% (30/183) of answers judged as exact were retrieved using the knowledge annotation paradigm. We believe that such a performance is remarkable, considering that our system contained only twenty-eight database access schemata with access to seven knowledge sources. In total, the knowledge annotation component represented no more than a few person-days worth of manual labor.

These results also verify that analysis of typical user question distribution can help guide the knowledge engineering effort. Our database access schemata were geared towards answering the most frequently occurring questions from the previous TREC evaluations; many of the same question types also appeared in the 2002 testset.

7. LIMITATIONS

Despite our efforts in streamlining the knowledge engineering process for structuring Web resources, the need for manual labor remains the biggest limitation of the knowledge annotation approach to question answering. In our approach, database queries must be mapped to procedures capable of extracting only the relevant fragments of Web pages (using wrappers), and natural language queries must be mapped to specific database queries (using annotations). Because each site contains inconsistencies and idiosyncrasies, human effort is required to achieve high precision. To address the wrapper generation problem, we believe that machine learning techniques for automatically or semi-automatically inducing wrappers [21, 9, 13, 26] are promising. For crafting the mappings from natural language to database queries, we believe that developments in the automatic recognition of paraphrases [2, 14] will help systems equate questions that are asking for the same information.

Although our knowledge mining paradigm can achieve remarkable performance using only simple techniques, there remain difficult problems that only a deeper understanding of language can solve. One such problem is associated with questions whose answers are temporally dependent, e.g., "Who was the prime minister of Britain in 1979?" or "Who was the first governor of Missouri?" Since Aranea does not recognize temporal expressions, it usually answers with the *current* prime minister and *current* governor. Another set of problems center around the semantic importance of modifiers, e.g., "Who was the *second* man to set foot on the moon?" or "What is the *third* tallest peak in the world?" In these questions, the keywords "second" and "third" are extremely important semantically; however, Aranea considers them statistically unimportant (because they appear very frequently on the Web and in corpora). As a result, our system often ignores the constraints imposed by those modifiers; i.e., Aranea might answer the above questions with "Neil Armstrong" and "Mt. Everest", respectively. We believe that integrating linguistically-sophisticated techniques into Aranea is the solution to these problems (cf. [17]). However, the effective integration of precise, but brittle natural language processing technology with robust, but less precise statistical techniques remains a research challenge.

8. CONTRIBUTIONS

The Aranea system implements and integrates two different paradigms for answering open-domain factoid questions. Answering questions can be viewed as executing database queries using our knowledge annotation technique. Alternatively, answers to questions could be "mined" from the Web using knowledge mining techniques. A primary contribution of Aranea is the smooth integration of these two approaches into a uniform framework. Because the distribution of factoid questions roughly follows Zipf's Law, we can employ knowledge annotation techniques to handle the "head" of the curve and utilize knowledge mining techniques to handle its "tail."

Another salient aspect of the Aranea approach is a task-driven strategy. Since the ultimate goal of a question answering system is to answer user questions, it is only natural to analyze *what users actually ask*, in the form of logs and various testsets. By doing exactly this, we formed a characterization of real-world user queries, which enabled the smooth integration of two very different approaches to question answering.

9. REFERENCES

[1] Paolo Atzeni, Giansalvatore Mecca, and Paolo Merialdo. Semistructured and structured data in the Web: Going back and forth. In *Proceedings of the Workshop on Management of Semistructured Data at PODS/SIGMOD'97*, 1997.

[2] Regina Barzilay and Kathleen McKeown. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-2001)*, 2001.

[3] Eric Breck, Marc Light, Gideon S. Mann, Ellen Riloff, Brianne Brown, Pranav Anand, Mats Rooth, and Michael Thelen. Looking under the hood: Tools for

diagnosing your question answering engine. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-2001) Workshop on Open-Domain Question Answering*, 2001.

[4] Eric Brill, Susan Dumais, and Michele Banko. An analysis of the AskMSR question-answering system. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, 2002.

[5] Eric Brill, Jimmy Lin, Michele Banko, Susan Dumais, and Andrew Ng. Data-intensive question answering. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001)*, 2001.

[6] Sabine Buchholz. Using grammatical relations, answer frequencies and the World Wide Web for question answering. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001)*, 2001.

[7] Charles Clarke, Gordon Cormack, and Thomas Lynam. Exploiting redundancy in question answering. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-2001)*, 2001.

[8] William Cohen, Andrew McCallum, and Dallan Quass. Learning to understand the Web. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 23:17–24, 2000.

[9] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam, and Sean Slattery. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-1998)*, 1998.

[10] Daniela Florescu, Alon Levy, and Alberto Mendelzon. Database techniques for the World-Wide Web: A survey. *SIGMOD Record*, 27(3):59–74, 1998.

[11] Joachim Hammer, Hector Garcia-Molina, Junghoo Cho, Rohan Aranha, and Arturo Crespo. Extracting semistructured information from the Web. In *Proceedings of the Workshop on Management of Semistructured Data at PODS/SIGMOD'97*, 1997.

[12] Eduard Hovy, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. Using knowledge to facilitate factoid answer pinpointing. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002)*, 2002.

[13] Chun-Nan Hsu and Chien-Chi Chang. Finite-state transducers for semi-structured text mining. In *Proceedings of the IJCAI-99 Workshop on Text Mining: Foundations, Techniques, and Applications*, 1999.

[14] Ali Ibrahim, Boris Katz, and Jimmy Lin. Extracting structural paraphrases from aligned monolingual corpora. In *Proceedings of the Second International Workshop on Paraphrasing (IWP-2003)*, 2003.

[15] Boris Katz. Annotating the World Wide Web using natural language. In *Proceedings of the 5th RIAO Conference on Computer Assisted Information Searching on the Internet (RIAO '97)*, 1997.

[16] Boris Katz, Sue Felshin, Deniz Yuret, Ali Ibrahim, Jimmy Lin, Gregory Marton, Alton Jerome McFarland, and Baris Temelkuran. Omnibase: Uniform access to heterogeneous data for question answering. In *Proceedings of the 7th International Workshop on Applications of Natural Language to Information Systems (NLDB 2002)*, 2002.

[17] Boris Katz and Jimmy Lin. Selectively using relations to improve precision in question answering. In *Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering*, 2003.

[18] Boris Katz and Patrick Winston. Method and apparatus for generating and utilizing annotations to facilitate computer text retrieval, United States Patent No. 5,309,359, 1994.

[19] Boris Katz and Patrick Winston. Method and apparatus for utilizing annotations to facilitate computer retrieval of database material, United States Patent No. 5,404,295, 1995.

[20] Craig Knoblock, Steven Minton, Jose Luis Ambite, Naveen Ashish, Ion Muslea, Andrew Philpot, and Sheila Tejada. The Ariadne approach to Web-based information integration. *International Journal on Cooperative Information Systems (IJCIS) Special Issue on Intelligent Information Agents: Theory and Applications*, 10(1/2):145–169, 2001.

[21] Nickolas Kushmerick, Daniel Weld, and Robert Doorenbos. Wrapper induction for information extraction. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, 1997.

[22] Cody Kwok, Oren Etzioni, and Daniel S. Weld. Scaling question answering to the Web. In *Proceedings of the Tenth International World Wide Web Conference (WWW10)*, 2001.

[23] Jimmy Lin. The Web as a resource for question answering: Perspectives and challenges. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-2002)*, 2002.

[24] Jimmy Lin, Dennis Quan, Vineet Sinha, Karun Bakshi, David Huynh, Boris Katz, and David R. Karger. The role of context in question answering systems. In *Proceedings of the 2003 SIGCHI Conference on Human Factors in Computing Systems (CHI 2003)*, 2003.

[25] John B. Lowe. What's in store for question answering? (invited talk). In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*, 2000.

[26] Ion Muslea, Steve Minton, and Craig Knoblock. A hierarchical approach to wrapper induction. In *Proceedings of the 3rd International Conference on Autonomous Agents*, 1999.

[27] Ellen M. Voorhees. Overview of the TREC 2001 question answering track. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001)*, 2001.

[28] Ellen M. Voorhees. Overview of the TREC 2002 question answering track. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, 2002.

[29] Ellen M. Voorhees and Dawn M. Tice. Overview of the TREC-9 question answering track. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, 2000.