Answer Verification for Improved Precision in a Web-Based Question Answering System

by

Wesley A. Hildebrandt

Submitted to the
Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2003

Author .....
Department of Electrical Engineering and Computer Science
May 21, 2003

Certified by.....
Boris Katz
Principal Research Scientist
Thesis Supervisor

Accepted by ..
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Answer Verification for Improved Precision in a Web-Based Question Answering System

by

Wesley A. Hildebrandt

Submitted to the Department of Electrical Engineering and Computer Science
on May 21, 2003, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

VOCAULA, for Verification Of Candidate Answers Using Linguistic Analysis, is an answer verification system and methodology developed as a component for a larger Web-based question answering system, Aranea. Aranea uses the high recall available by applying information retrieval techniques to the World Wide Web to increase domain coverage. This approach generates multiple candidate answers for any question; VOCAULA picks the best of these candidates by considering linguistic constraints present in the question and comparing these to supporting information for the candidate answers. Although intended for a particular system, this component's modular design allows it to easily integrate into any question answering system which is based on high recall from large corpora. In evaluation against a set of TREC test questions, VOCAULA increased the number of correct answers by nearly ten percent over the base Aranea system. This is a modest gain, but it shows that answer verification should play a part in any question answering system using an information retrieval focused approach.

Thesis Supervisor: Boris Katz
Title: Principal Research Scientist

Acknowledgments

This thesis would not have been possible without the incredible love and support of my wife Laura. She and my kids Charlie and Emma have had to put up with many nights when I've been working on this thesis instead of spending time with them.

My parents always taught me I could do anything I set my mind to, and they have always been there for me. Thank you, Mom and Dad!

Too many people have encouraged my love of learning for me to thank them individually, but to all the teachers, instructors, and professors who have influenced me throughout my education I'd like to express my heartfelt thanks.

For this thesis and the research leading up to it, I am particularly grateful to Boris Katz and the members of his InfoLab group at MIT's Artificial Intelligence Laboratory. For their comments, pointers, and good conversations over the years, I would like to thank Deniz Yuret, Michael de la Maza, Sue Felshin, Jimmy Lin, and Greg Marton.

In closing, I must thank science fiction writers everywhere. They help us dream of an ever-improving future in which technology increases human access to knowledge in an increasingly unobtrusive manner. As FDR once said, "The only limit to our realization of tomorrow will be our doubts of today. Let us move forward with strong and active faith."

This material is based upon work supported under a National Science Foundation Graduate Research Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this thesis are those of the author and do not necessarily reflect the views of the National Science Foundation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Ever since there was an Oracle at Delphi, mankind has dreamt of a knowledge source that always has the right answer and is conversant on any subject. Over the last decade, the explosion of the World Wide Web has provided electronic access to this level of universal information, but actually finding the exact answer one is looking for has turned out to resemble the proverbial search for a needle in a haystack. This thesis describes a component that can be used with a Web-based question answering system to improve the precision of the answers returned to the user. VOCAULA, for Verification of Candidate Answers Using Linguistic Analysis, applies *partial parsing* of questions, candidate answers, and supporting information to the task of deciding which single answer is the best one to offer in response to a query. Since complete parsing of open text (especially text found on the Web) is still beyond the state of the art, partial parsing is used to recover much of the linguistic knowledge that is available while avoiding the difficulty of full parsing.

1.1 Background

There are two primary approaches in general use by question answering systems. The linguistically focused approach begins by parsing the input question and comparing it to a knowledge database. This database may contain individual facts, pointers to larger text segments or multimedia answers, methods of constructing answers to

specific question types, or the results of limited parsing of corpora. Despite decades of research, open text parsing remains beyond the state of the art, which means that the knowledge database used by this approach requires significant human development and maintenance. One example of this type of question answering system, based entirely on natural language parsing, is the START system, developed by Boris Katz and his InfoLab group at MIT's Artificial Intelligence Laboratory over the last two decades [13, 14]. Since 1993, START has answered hundreds of thousands of users' questions on many topics via a web portal [11]. Despite their successes, the addition of new knowledge to systems like START remains a manpower-intensive task.

To overcome this limitation, an alternative approach focused on information retrieval has become popular in recent years. Contributing to its popularity is the ready access to very large corpora which are continually updated, along with an ability to easily search those corpora for particular words and phrases. Specifically, the advent of the World Wide Web and associated search engines (e.g., Google™) have forever changed the landscape of knowledge retrieval and question answering systems. By using the Web and the several billion pages currently indexed by search engines, information on virtually any topic can be found.

However, this alternative approach also has its drawbacks. Treating the input query as a set of words or simple phrases discards information about the relationships between those words and phrases. This information is often vital to ensuring the answer given actually answers the question asked. For example, the questions "When did Germany invade Poland?" and "When did Poland invade Germany?" contain exactly the same words, but are referring to two different events (one of which has not happened).

Answer verification is the process of taking potential answers returned by a high-recall system (usually based on keyword searching), and increasing the precision of those answers using a linguistically motivated analysis of the questions and answers. This thesis describes VOCAULA, an answer verification system and methodology developed as a component for a larger Web-based question answering system, Aranea [18]. Although it was intended for a particular system, this component's modular de-

sign will allow it to be quickly integrated into any similar question answering system. When a set of test questions from the Text REtrieval Conference (TREC) Question Answering (QA) tracks [33, 38] were used to compare the performance of the base Aranea system against a VOCAULA-enhanced system, VOCAULA increased the number of correctly answered questions by almost ten percent. This is a modest gain, but it shows that answer verification can be a useful component of any question answering system using an information retrieval focused approach.

1.2 Thesis Outline

This thesis is organized as follows. Chapter 2 presents some background on the question answering task that has led us to where we are today, and Chapter 3 describes the Aranea system the VOCAULA component was designed to integrate with. Chapters 4 and 5 discuss the VOCAULA system in detail, as well as other related and similar research. Chapters 6 through 8 show results of testing this system against different question sets, discuss some of the successes and failures of this system, and offer avenues for further improvement of the system. Finally, Chapter 9 closes with a high-level summary of observations on the VOCAULA system.

Chapter 2

Question Answering History

Ultimately, question answering is the task of providing information to a user who requests it. The typical input is a natural language question, English for many systems, and the output is some set of data that contains the answer to that question. The goal is to be able to respond with exactly the facts the user needs, with perfect accuracy and without extraneous information. The state of the art can not yet achieve this on open domain questions, but we are much closer than just a few years ago. The following is an overview history of the computer-assisted search for information. The ordering of these topics is not intended to imply that each one strictly precedes the next, or that they are in any way mutually exclusive. Each field is useful in its own right, and the progression presented here merely shows an increase in the similarity to current (and future) question answering systems.

2.1 Information Retrieval

The oldest form of electronic question answering is information retrieval, in which the task is to find those documents in a large collection which are most relevant to a user's query. Formal conferences on information retrieval have been held for at least three decades [26], and much progress has been made, especially in applying techniques to the Web [1] and adapting to changing user query habits [27]. However, these systems are primarily designed to return entire documents (or links to documents), which is

generally more information than the user wants or needs. Superimposed methods to reduce the size of the returned "answer" typically involve passage ranking algorithms using keywords in the query [15], and still do not provide an exact answer to a question.

2.2 Information Extraction

One step closer to modern question answering systems is information extraction, which is associated with predefined schemata whose slots are filled with data from documents in a corpus. A question is matched against the appropriate schema, and portions of the correct filled-in schema are returned as an answer. This approach to fulfilling users' knowledge needs was championed by the TIPSTER Text Program [32] and the Message Understanding Conferences [7, 30]. Because this method requires computationally-intensive preprocessing of an entire corpus, it is not well suited for today's large corpora (an extreme example of which is the World Wide Web). It is also limited to the specific question types for which schemata have been developed and instantiated with information from the corpus. It will never be possible to write schemata for every type of question a user may ask, and the required preprocessing of the corpus makes it difficult to add a schema after the fact.

2.3 Relational Database Access

This method of question answering uses a set of rules to translate users' questions into queries against a classic relational database (e.g., LUNAR [39]). For questions whose answers are contained in the database, this method is highly reliable and can return an exact answer to the user. However, in many cases users are unclear what knowledge they can ask about (i.e., what the limits of the database are), or assume that since the system uses natural language it must be "smart," so they do not have to submit as precise a query as they would otherwise [9]. Additionally, users are sometimes unclear as to whether a failure of the system was caused by a lack of coverage in the

database, a misunderstanding of the question, or a conceptual mismatch between the question and the database.

2.4 Question Answering

Question answering has the goal of combining the best features of each of these approaches to create a single universal system that has wide topic coverage and returns to the user the correct information he is looking for, without leaving pertinent details out or including extraneous information. Over the last decade the Text REtrieval Conferences (TREC), co-sponsored by the National Institute of Standards and Technology (NIST), have focused research in question answering and contributed to significant advances in the state of the art [35]. A combination of various approaches can now provide exact, correct answers in many limited domains, but delivering this performance on open domain questions is still an outstanding research problem. VOCAULA aims to take another step toward this ideal, by taking the output of a high recall question answering system designed for maximum coverage of open domain topics, and using linguistic analysis to improve the accuracy and relevance of the answers returned to the users.

Chapter 3

Aranea Overview

VOCAULA was designed to work within the Aranea framework [18], specifically within the Knowledge Mining module. However, it has been packaged in such a way that it could be easily integrated into any Web-based question answering system. Aranea is a Web-based question answering system developed by the InfoLab group at MIT, and was submitted to the question answering track of TREC-11 [34]. Aranea uses two Web-based methods of discovering answers to users' questions, along with a postprocessing pipeline designed to improve the accuracy of the answers and project those answers into the AQUAINT corpus used for the TREC-11 evaluation. Figure 3-1 shows an outline of the Aranea framework.

The answer generation portion of Aranea is split into Knowledge Annotation and Knowledge Mining modules because of the empirical observation that although many questions users ask can be categorized into a few question types, the rest of the questions tend to be unique. This Zipf's Law-like behavior [40] has been observed in formal question lists such as those used in the TREC question answering track [17], as well as in more informal question collections such as logs from the natural language processing-based question answering system START [10] and a web search engine [20]. It is interesting to note here that even when a search engine is not specifically designed to handle only natural language questions, a significant number of users still tend to query the system exclusively with complete questions. This once again demonstrates that many users find natural language questions to be the easiest way to query a

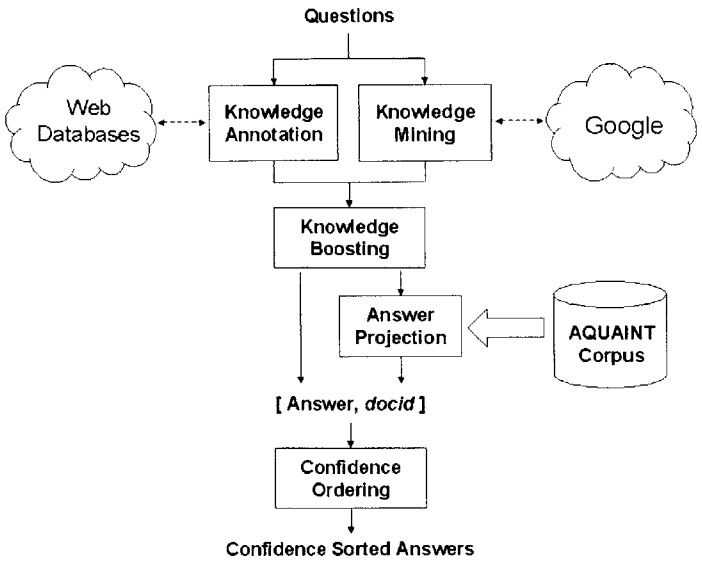

Figure 3-1: Aranea organization and data flow

knowledge retrieval system.

3.1 Knowledge Annotation

The first module used by Aranea to find answers on the Web is called Knowledge Annotation. Knowledge Annotation joins many databases of information that are available on the Web under a common interface, similar to the Omnibase system used by START [12]. These web databases have the answers to many question types stored in semi-regular structures that can be automatically parsed. For example, questions about countries (e.g., population, size, and languages) can be answered with data from the CIA's World Factbook [6], and answers to questions about the Messier deep space objects (e.g., location, brightness, and distance) could be found on a website hosted by Students for the Exploration and Development of Space [28, 29]. The two key factors that allow these websites to be used by the Knowledge Annotation module are (1) the data they each contain is in some kind of regular structure such as a table or labeled paragraphs, and (2) data for a particular entity can be accessed with an automatically crafted HyperText Transfer Protocol (HTTP) request. For each website that is to be used as a back-end, and for each type of question that site should be able to answer, a

small amount of manual schema development is needed. These schemata allow users' questions to be converted into properly formatted HTTP requests, and specify how the returned web page is to be parsed to recover the answer of interest. Although there is some human involvement required to initially generate these schemata (as well as periodic maintenance required if websites change the format of their pages), the payback is well worth it. Schemata are only constructed for questions on the initial, high-frequency portion of the Zipf's curve of question types, so each schema constructed allows the system to answer a large number of user questions. For TREC-11, Aranea used schemata for only seven sites in its Knowledge Annotation module, but even so fifteen percent of its correct answers came from this module.

3.2 Knowledge Mining

The other method Aranea uses to find answers on the Web is called Knowledge Mining. Figure 3-2 shows the internal workings of this module and how VOCAULA fits into its structure. Knowledge Mining relies on the incredible size of the Web to search for simple patterns that contain the question and answer, and falls back on keyword searching if necessary. Since the Web is several orders of magnitude larger than any other electronic corpus, there are usually multiple locations with the answer to any given question, making it easier to find that answer. In fact, Breck et al. [3] showed that there is a strong correlation between the number of times an answer appeared in the TREC-8 corpus [37] and the average performance of systems on questions for which that many answers appeared.

The answer redundancy available on the Web in many cases obviates the need for complicated natural language processing. Here is an actual example from the Web that illustrates this point:

> (Q) Who was the last man on the moon?
>
> (A1) Flight Commander Eugene Cernan was the last man on the moon.
>
> (A2) Captain Cernan has logged 566 hours and 15 minutes in space, of which more than 73 hours were spent on the surface of the moon, and was the last man of Apollo to leave his footprints on the surface of the moon.

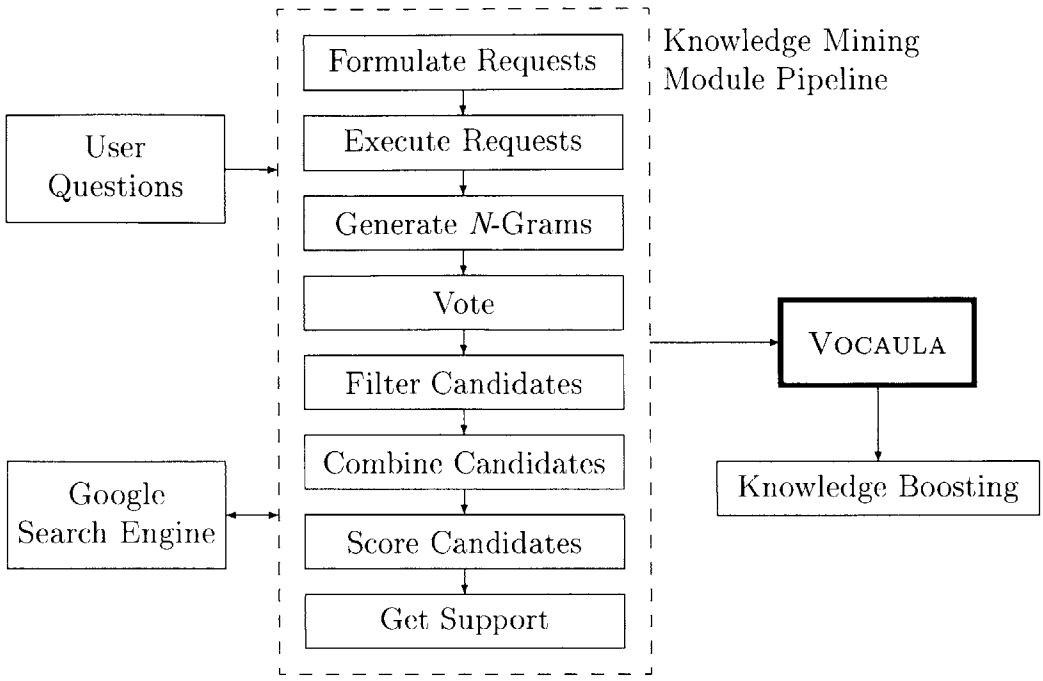

Figure 3-2: VOCAULA integration into the Knowledge Mining module

Both sentences A1 and A2 contain the answer to the question, and both are direct quotes from web pages accessible through a standard search engine [23, 31]. However, it is clear that deriving the answer will be much easier using A1. With a smaller corpus, the only information available might be A2, which would either require sophisticated natural language processing and semantic understanding or would force a keyword-type search.

Although it seems that this data redundancy could be the cure question answering systems have been searching for, there are problems with data derived from the Web that reduce the overall effectiveness of this approach. First, there is no editorial requirement prior to "publishing" information on the Web, so the quality of individual sites and documents varies widely. Some sites are inherently more reliable than average (e.g., CNN.com or IRS.gov), but there is no method currently available to measure this reliability. In addition, even the most reliable sources can be mistaken at times (consider the BBC article which reprinted the urban legend of a man who had been dead at his desk for five days before coworkers noticed [2]). Since the reliability of any single source must be questioned, multiple sources should be polled to

confirm that they agree. Of course, the obvious way to apply this reliability check is to choose the answer reported by the highest number of independent sources, which is not necessarily a guarantee of correctness.

Even for those sites which contain accurate information, the general lack of editing means typographical and grammatical errors are common, and proper punctuation is intermittent at best. "Sophisticated" parsing methods often break when confronted with these types of inaccuracies, requiring a fallback method. The following is a brief outline of the steps the Knowledge Mining module takes in applying these considerations to generate candidate answers from a user's question.

3.2.1 Formulating and Executing Queries

Since web search engines are used as the basis for information recall in this module, the first step is to turn the user's question into queries suitable for the particular search engine being used. Currently, Google™ is the search engine of choice, primarily because it has wide coverage of the Web and it returns text "snippets" from the web pages that match the query. These snippets allow an answer to be determined for many questions without ever accessing the source pages, which speeds up the question answering process and avoids links to web pages that are not immediately accessible.

Two types of queries are generated in the Formulate Requests step. The first type is simply a "bag of words" query, and is included as a fallback method when more exact searches do not return (enough) hits. The second type, *exact queries*, use pattern matching rules on the question to generate potential phrases that will contain the answer. For example, one of the conversions for the question "What did Eli Whitney invent?" is the pattern "Eli Whitney invented X." In this case additional constraints are imposed on X: it must occur following the exact phrase "Eli Whitney invented," and it must be within a specified number of words or characters of the end of that phrase. Each question generates at least the keyword query, and also as many exact queries as there are pattern matching rules that apply to the question.

The Execute Requests step then creates the proper HTTP commands to send the generated queries to the search engine, and parses the returned pages to extract the

text snippets. For keyword searches all snippets are used. Since some constraints imposed on exact queries cannot be enforced when sending the request to the search engine, postprocessing is done on the snippets returned by those queries. Only the snippets that meet all constraints are passed on to the next step.

3.2.2 Generating and Scoring *N*-Grams

In the Generate *N*-Grams step, all series of consecutive words up to four words in length are extracted from the snippets generated by the first two steps. Each of these "phrases" is given a score based on the query which generated the snippet that phrase was located in, with exact queries generally weighted higher than keyword queries. Next, the Vote step scores each unique n-gram by summing the scores for all copies of it. In essence, this weights more frequent phrases higher.

3.2.3 Filtering Candidates

The phrases generated by the preceding steps are the initial answer candidates. To reduce the size of this list, some candidates are culled from the list during the Filter Candidates step. In particular, numerical answers to *who* and *where* questions and non-numerical answers to *how far*, *how fast*, *how tall*, etc., are eliminated. This is based on the empirical observation that certain question types require a numerical answer, and others prohibit one. In addition, candidates that begin or end with stop words are removed, since shorter answers without the stop words would also have been found during the previous steps, and the stop words do not contribute to the correctness of the answer. Finally, most candidates that contain a keyword from the question are removed. The exception is questions like "What kind of bridge is X?," for which "suspension bridge" is an acceptable answer. The overall goal of this step is to remove only those candidates that are definitely not potential answers, ensuring that correct answers are not inadvertently discarded at this stage. It is left for follow-on steps and modules to further intelligently reduce and then prioritize this field of potential candidates.

3.2.4 Normalizing Candidates and Checking for Support

The first step in normalizing candidates is to Combine Candidates. Candidates are combined by adding the score of a shorter candidate to any longer candidate that completely contains the shorter one as a substring, and then eliminating the shorter candidate from consideration. This favors longer candidates, opposing the inclination of the n-gram scoring method to favor shorter answers simply because they occur more often. The second step in normalizing candidates is to Score Candidates. Here, the score for each candidate answer is scaled relative to the a priori distribution of its component words, to ensure that answers using relatively common words are not unfairly promoted over those with less common words. The final step in the classic Knowledge Mining pipeline is to Get Support. This step checks the output of the previous steps, and removes any candidate answers that do not actually appear in the original snippets.

3.3 VOCAULA Integration

After the Knowledge Mining module prepares a list of candidate answers, the base Aranea system passes that list to the Knowledge Boosting module. VOCAULA is designed to sit between these two modules, and reorders the candidate answers using linguistic analysis of the question, the candidate answers, and the snippets that support each answer. Chapter 4 describes this system in detail.

3.4 Knowledge Boosting

The Knowledge Boosting module uses part-of-speech information and pattern matching to check that candidate answers are complete phrases, and removes any extraneous words before or after the phrase. It also uses some heuristics to increase the score of certain candidate answers for particular types of questions. For example, candidate answers for questions that require a geographical body for an answer (e.g., *what city/state/country ...*) are compared against an external list of such entities, and

answers on the list are promoted. Also, a date recognizer is used to promote answers that meet its criteria for questions that require a date answer (e.g., *what year* ...). If the question asks for a year, the date answers are also simplified by removing month and day information if it is present.

3.5 Answer Projection

Once a set of candidate answers is decided on, supporting documents from the AQUAINT corpus must be found in order to meet the criteria of the TREC evaluations. Three different methods are used by Aranea to project the candidate answers into the corpus, differing in how the passage retrieval algorithm behaves. Along with a document number for each answer, this module also produces a score indicating the confidence that the document chosen truly does support the answer.

3.6 Confidence Ordering

For TREC-11, each submitted run was scored based not only on how many questions were answered correctly, but also on how the questions were ordered, with correct (or incorrect!) answers toward the beginning of the list having more weight than those toward the end. Aranea uses simple heuristics to improve its performance relative to this scoring methodology, ranking questions by type in the following four general groups: first *when*, followed by *who* and *where*, then *what*, and finally all others. Within these four groups, questions whose answers came from the Knowledge Annotation module rank higher than those whose answers come from the Knowledge Mining module, and then questions are ranked by the Answer Projection score achieved by the answer and document number pair. Any remaining ties are arbitrated by the Knowledge Mining score.

3.7 TREC-11 Results

Aranea performed fairly well at TREC-11, but two major areas were noted for improvement. First, if projection of the answers into the AQUAINT corpus was not required, the number of correct answers increased by 20 percent. Second, the Knowledge Mining component generated an incorrect answer for more than two-thirds of the questions it answered, while the Knowledge Annotation module was incorrect less than half the time. VOCAULA was developed to address this second issue, by boosting the number of correct answers generated by the Knowledge Mining module.

Chapter 4

VOCAULA Description

VOCAULA was designed to improve the precision of a high-recall question answering system which uses the Web as its information source. In particular, it was designed to work as part of the Aranea project, which was described in some detail in Chapter 3. VOCAULA takes the output of the Knowledge Mining portion of Aranea, and uses linguistic analysis to rerank the candidate answers. The input to VOCAULA is a list of questions, along with an ordered list of candidate answers to each question and supporting information for each of those candidate answers. In this case the supporting information takes the form of text "snippets," as returned by a web search engine (i.e., Google). The output of VOCAULA is a re-ordered list of the candidate answers, with the intent being to increase the probability of the top-ranked answer being correct.

4.1 Linguistic Analysis of Questions

The central component of this system is a partial parser, which performs limited linguistic analysis on both the question and the candidate answers with their supporting information. Since the format of the questions and answers is different, the analysis of each is performed differently. For questions, the partial parser is designed to understand many common question structures, and can "invert" these questions from interrogatives into declarative statements.

4.1.1 Preprocessing Input Text

The first task in analyzing each input question is to conduct some preprocessing of the interrogative to prepare it for tagging with part-of-speech tags. This preprocessing consists of six discrete steps.

First, all common contractions are expanded. This simplifies the rules for the partial parsing since no special rules are needed for contractions. Table 4.1 shows the list of contractions that are expanded. For the expansion of *n't* to *not*, the root word is also changed if necessary (e.g., *won't* $\Rightarrow$ *will not*). Notice that *'d* is expanded as *would*, although it could also mean *had*. These two can be distinguished by considering the inflection(s) on later words in the verb phrase. For example, in *He'd eaten the apple*, the past participle inflection on *eat* forces *'d* to be interpreted as *had*, while in *I asked if he'd eat the apple*, the uninflected form of *eat* forces the *would* interpretation of *'d*. Distinguishing these cases will be performed in a future version of this system.

n't $\Longrightarrow$ not	'd $\Longrightarrow$ would	'll $\Longrightarrow$ will
'm $\Longrightarrow$ am	're $\Longrightarrow$ are	've $\Longrightarrow$ have

Table 4.1: Expansion of common contractions

The next step is to expand *'s* contractions. This is separate from the first step because *'s* can also be used to indicate possessive as well as *is* (the *'s* suffix can also replace *has*, but this use is very infrequent). In fact, it is difficult to distinguish these two cases without parsing the entire sentence, so this rule is limited to those cases where the root word cannot have the possessive inflection *'s* (e.g., pronouns and question words like *who* or *what*).

The third step is to expand certain abbreviations. There are two types of abbreviations considered here: acronyms and shortened words. Since acronyms are almost always ambiguous without semantic contextual information, only a few acronyms with high probabilities are expanded (e.g., *US* $\Rightarrow$ *United States*). Shortened words are less ambiguous, especially as part of a title (e.g., *dr.* $\Rightarrow$ *doctor*, *jr.* $\Rightarrow$ *junior*). After abbreviations that can be safely expanded are processed, the next step is to normalize all remaining abbreviations. In general, this means that periods and extra

spacing are removed. For example, *U. S. N. A.* becomes *USNA*.

The fifth preprocessing step addresses capitalization since capitalization is used to help locate proper nouns during part-of-speech tagging. Specifically, the first word in the question is changed to lower case unless it might be part of a proper name. In order to reduce the complexity of the partial parser, the final preprocessing step joins all text within quotes into a single "word" for part-of-speech tagging and parsing.

4.1.2 Part-of-Speech Tagging

After the questions are preprocessed, part-of-speech tags are assigned using a Perl module based on Brill's rule-based part-of-speech tagger [4]. Once tags are assigned several rules are applied that correct problems with the tagger and prepare the sentence for parsing. The first group of rules corrects some instances of mistagged words:

- Adjectives mistagged as nouns – e.g., *official* in *the official first day of summer*
- Nouns mistagged as prepositions – e.g., *inside* in *the inside of the box*
- Verbs mistagged as nouns – e.g., *works* in *who works on Capitol Hill*
- Adverbs mistagged as adjectives – e.g., *first* in *first, I had breakfast*

The next rules join together some multiword verbs, which take two forms: verb + verbal preposition(s), and verb + infinitive. Examples of common multiword verbs using verbal prepositions are *turn around*, *speak up*, and *run away from*; an example of a verb joined with an infinitive is *have to be*.

There is also a single rule in this section to solve one type of anaphoric reference. If a question begins with a noun phrase, followed by a question with an intrasentential anaphoric reference in it, the noun phrase replaces the reference. For example, "Mercury, what year was it discovered?" is transformed to "what year was Mercury discovered?".

The final step is to normalize monetary phrases by turning symbols into words (e.g., \$ $\Rightarrow$ *dollar*) and to remove any remaining symbols that are not understood.

4.1.3 Question Pattern Matching

Now that the questions have been put in a normalized form with part-of-speech tags assigned, they are checked against a series of patterns to conduct partial parsing of the questions and extract some of the linguistic information. These patterns were hand-coded, based on a set of training questions. Table 4.2 shows the patterns used by this version of the system. Each of these patterns has some flexibility to allow for inflections, adverbial phrases, etc. The placeholders ⟨NP⟩ and ⟨PP⟩ stand for the typical noun phrase and prepositional phrase, respectively, but ⟨VP⟩ only represents the main verb along with any auxiliary verbs. ⟨aux⟩ stands for any auxiliary verb, ⟨prep⟩ is a preposition, ⟨extent⟩ is any complement to a *how* question (e.g., *long*, *fast*, *far away*), and ... is a (possibly empty) sequence of words. Additionally, words like *is* can match other inflections of the word, such as *are* or *were*.

After a question is matched against a particular pattern, linguistic information such as subject, verb, and object is extracted and used to create declarative statements that, if true, could verify the accuracy of a candidate answer. These statements are similar in construction to the "validation statements" created by DIOGENE (see Section 5.3), except that DIOGENE does not actually use the statements for answer validation. The developers of DIOGENE note that the statements themselves are too strict to find many matches, so they relax the statements into validation patterns (i.e., keyword matching). VOCAULA does not need to find new answers to questions, but only checks to see if some answer not ranked at the top of the list by the Knowledge Mining module in Aranea should be promoted. In the context of answer verification, not matching an exact validation statement means that there is less chance that a lower-ranked answer with the right supporting information will be promoted. However, it also means that there is less chance that a incorrect lower-ranked answer will be promoted simply because some keywords match between its supporting information and the validation statement(s).

Definitional Questions	
What is ⟨NP⟩	Name ...
What is the name of ...	What do you call ...

Question Word (Q-word), not followed by ⟨NP⟩	
Q-word [⟨PP⟩] ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...	Q-word [⟨PP⟩] ⟨aux⟩ ⟨VP⟩ ...
Q-word [⟨PP⟩] are ⟨NP⟩ ⟨NP⟩ ⟨prep⟩	Q-word [⟨PP⟩] ⟨verb⟩ ...

What / Which ⟨NP⟩ ...	
What/Which ⟨NP⟩ ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ⟨prep⟩ ...	What/Which ⟨NP⟩ ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...
What/Which ⟨NP⟩ is ⟨NP⟩ ⟨NP⟩ ⟨prep⟩	What/Which ⟨NP⟩'s ⟨NP⟩ is ⟨NP⟩
What/Which ⟨NP⟩ ⟨aux⟩ ⟨VP⟩ ...	What/Which ⟨NP⟩ is ⟨NP⟩ ⟨prep⟩
What/Which ⟨NP⟩ is ⟨NP⟩	What/Which ⟨NP⟩ are there ⟨PP⟩
What/Which ⟨NP⟩ ⟨VP⟩ ⟨NP⟩	What/Which ⟨NP⟩ ...

How ... Questions	
How many ⟨NP⟩/⟨PP⟩ per ⟨NP⟩	How many ⟨NP⟩/⟨PP⟩ are there
How many ... ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...	How many ⟨NP⟩/⟨PP⟩ ⟨VP⟩ ...
How much ⟨NP⟩ ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...	How much ⟨PP⟩ ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...
How much ⟨NP⟩ ⟨VP⟩ ...	How much ⟨PP⟩ ⟨VP⟩ ...
How much ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...	How ⟨extent⟩ ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...
How ⟨extent⟩ is ⟨NP⟩ ⟨PP⟩	How ⟨extent⟩ ⟨VP⟩ ...
How ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...	How is ⟨NP⟩

Miscellaneous Question Patterns	
⟨prep⟩ what/which ⟨NP⟩ ⟨aux⟩ ⟨NP⟩ ⟨VP⟩ ...	⟨prep⟩ what/which ⟨NP⟩ is ⟨NP⟩ ...
⟨prep⟩ how ⟨extent⟩ ⟨aux⟩ ⟨NP⟩ ...	... what/which ⟨NP⟩
Could you say what/which ⟨NP⟩ ...	Where is ⟨NP⟩ located

Table 4.2: Question patterns for partial parsing

4.2 Supporting Information

As mentioned previously, each candidate answer for each question is presented along with "supporting" information. This information is a list of the snippets returned by Google which the Knowledge Mining module used as evidence to generate this candidate answer. Text on the Web is generally not edited to the same extent as written material, so grammatical, spelling, and punctuation errors are common. In addition, each snippet returned by Google is a collection of multiple text segments from the underlying web page joined together with ellipses. These segments are not necessarily terminated at sentence boundaries. Because of these limitations, no standard parser can parse web snippets.

4.2.1 Preparing Text Snippets

Since classical parsing of text snippets is not possible, other methods must be used to check if a particular snippet meets the linguistic constraints imposed by a question. In VOCAULA, this is accomplished by comparing the snippet to inversions of the question, instantiated with the candidate answer. Before this can occur, the snippets must be prepared so they are in a normalized form that can be compared directly to the inverted questions. The snippets are first separated into the original text segments as delineated by ellipses, and each piece is then prepared similarly to the way the questions were prepared. Specifically, the preprocessing described in Section 4.1.1 and part-of-speech tagging is performed on each piece of the snippet.

4.2.2 Evaluating Answer Support

After preparation is complete, each segment of each snippet is compared to the question inversion(s) to see if that segment supports the candidate answer. In order to do this, the question inversion must be instantiated with the candidate answer. When the question inversions are generated (per section 4.1.3), a placeholder is included that indicates where in the inverted statement the candidate answer should be found. This placeholder is replaced with the candidate answer which the current snippet

supposedly supports, and then the instantiated inversion is compared to the snippet segment.

4.3 Ranking Candidate Answers

After the snippet segments for each candidate answer are compared against all instantiated inversions for that question, the candidate answers are (potentially) reranked using the following rules:

- If no match was found between any snippet segment and any instantiated inversion, the candidate answers are not reordered, and the highest ranking candidate as supplied by the Knowledge Mining module is returned to the user.
- If a single match is found, the candidate answer which matched is moved to the top of the list and is returned to the user.
- If multiple matches are found, all candidate answers with matches are ranked above candidates without matches, and within those two groups the original ordering is preserved.

These rules implement the following intuitions, which have also been empirically verified. First, candidate answers ranked higher by the Knowledge Mining module are more likely to be correct. This is a result of the keyword searching performed by that module, and is the basis of any information retrieval engine. Second, candidate answers which have matches between their supporting snippets and question inversions are more likely to be correct. For example, the question "How many Great Lakes are there?" with the candidate answer "five" yields the instantiated inversion "there are five Great Lakes." If one of the supporting snippets for the answer "five" contains the text "... that there are five Great Lakes and these are called ...," the answer "five" is clearly more likely to be correct. Of course, this relies on the accuracy of the snippets, which can be difficult to verify. For example, if one of the candidate answers for the above question is "six" and one of its supporting snippets

contains "... that there are six Great Lakes is a fallacy promoted by Senator Leahy of Vermont, who wants Congress to designate Lake Champlain a Great Lake ...," this system will be misled. Additional partial parsing to identify cases like this where negation or hypothesizing invalidates a potential match is a future research area for this system.

Chapter 5

Related Research

The automatic search for and retrieval of knowledge in response to a request for information, whether it is called information retrieval, passage extraction, or question answering, has a long history. The current forefront of this effort is embodied in the question answering track of the annual Text REtrieval Conferences (TREC) [36]. In the most recent conference (TREC-11 in 2002), 34 groups submitted a total of 67 main task runs, in which the goal was to return a single correct, exact, supported, and responsive answer to each of 500 questions. The answers had to be found in a collection of about 3 gigabytes of news articles from various sources (the AQUAINT corpus), and an answer was assessed as supported only if the news article it referenced actually contained that answer [34, 35]. To be correct and exact, a response had to completely answer the question, without any extraneous information and without any part of the answer missing. Answers to quantitative questions that did not contain units were judged as non-responsive, as were answers that referenced a different version of a specific object or person from the version referred to in the question. For example, "six" is an unresponsive answer to "How long is a fathom?," although "six feet" would be acceptable. Also, "1052 feet" is an unresponsive answer to the question "How tall is the Eiffel Tower in Epcot?," but it would be correct for the question "How tall is the Eiffel Tower in France?" Finally, the response "NIL" was judged as correct if (and only if) the document collection did not contain an answer to that question.

Several of the groups who participated in TREC-11 used some type of answer verification in their systems, although they did not always call it by that name. Also, some groups used the phrase "answer verification" to refer to different types of processing from what VOCAULA does. The following sections describe several system components that are either called answer verification components or perform a similar function.

5.1 IBM's T. J. Watson Research Center

For TREC-11, one IBM team submitted an update to their statistical question answering system, enhanced with a Web-based component they called "answer verification" [8]. Their system generated candidate answer sentences by searching the AQUAINT corpus with trained answer patterns for the particular question type. They then used statistical analysis to rank these sentences, and used a web search engine to provide two simple features to add to this analysis. The first feature they used was a binary feature indicating whether or not the candidate answer occurred in the top ten documents returned from a web query using the question as its basis, and the other feature was a count of the number of times the candidate answer was found overall on the Web.

One advantage this method has is that it is significantly less likely to generate candidate answers that are not present in the AQUAINT corpus, since the initial list of possible answer sentences is generated directly from the corpus. However, this method is less well suited for open domain question answering, in which the answers are not restricted to a single corpus. Additionally, the features derived from the Web are only used as part of the statistical analysis, and do not by themselves contribute to a significant performance improvement (they report only seven out of 140 correct answers are due to the Web-based features).

5.2 JAVELIN at Carnegie Mellon University

The JAVELIN system [25] submitted by Carnegie Mellon University to TREC-11 includes an Answer Generation module. One of the primary jobs of this module is to use different types of evidence to conduct answer verification. The only type of evidence currently implemented is the redundancy (or frequency count) of candidate answers, although plans call for future implementation of evidence from parse information [24]. This parse information will provide evidence in the form of matching high-level sentence structure between the question and answer (i.e., subject-verb-object agreement), as well as the recognition of internal answer invalidation, such as negation or hypothesizing. Matching high-level sentence structure is one of the key methods VOCAULA uses to verify candidate answers.

5.3 DIOGENE at the Istituto Trentino di Cultura

A team at the Istituto Trentino di Cultura's Center for Scientific and Technological Research (ITC-IRST) participated in TREC-11 with an update to their DIOGENE system, first used in TREC-10 [22]. One significant change to their system for TREC-11 was the addition of an answer validation component (described in more detail in [21]), which uses two web search methods to assess the truth of "validation statements." For example, the question "What is the top speed of a cougar?," combined with the candidate answer "50 miles per hour," yields the validation statement "The top speed of a cougar is 50 miles per hour." Since a specific statement may be too strict to find many results on the Web, they relax the validation statements into validation patterns, which are simply co-occurrences of the key terms in the question and answer that are restricted to occur close to each other in the same web page. Unlike VOCAULA, neither method they use to validate candidate answers retains any information about the relationships between the parts of the questions and answers.

5.4 Induction of Linguistic Knowledge Program

Buchholz, of the Induction of Linguistic Knowledge program at Tilburg University in the Netherlands, submitted a system called Shapaqa for evaluation during TREC-10 [5]. Shapaqa "chunks" each question into main verb, subject, object, and adjuncts, and then uses these chunks to match against sentences in a corpus, thereby maintaining information about the grammatical relationships in the question. One problem they experienced with the chunking of the questions was the question syntax itself. Since the partial parser in VOCAULA was written specifically for questions, it was able to resolve this issue.

Two version of Shapaqa were tested during the evaluation. The Shapaqa-TREC version matches chunked questions against sentences from the list of top documents supplied for each question by NIST, while the Shapaqa-WWW version does matching using a web search engine, then projects the answers back into the TREC corpus. This is similar to VOCAULA, except that VOCAULA does not apply grammatical constraints until after a set of potential answers has been generated by searching the Web. Applying grammatical constraints during the initial web search reduces the recall of a system (although possibly benefits precision). Their results confirm that the data redundancy in a larger corpus improves the reliability of question answering systems.

Chapter 6

System Evaluation

The VOCAULA system was trained and evaluated against questions from the TREC question answering tracks. Since its purpose is to improve the percentage of correct answers output from a Web-based question answering system, it was compared against the baseline Aranea system described in Chapter 3. Evaluating against the training and test question sets shows improvement over the baseline of twenty-one and eight percent, respectively.

6.1 Training and Test Question Sets

The initial set of questions used for training and testing VOCAULA came from the question answering tracks in TREC-9 and TREC-10 [33, 38]. Specifically, questions 201 through 700 from TREC-9 were used, since questions 701 through 893 were variation questions, asking the same thing as previous questions in different ways. Also, questions 894 through 1393 from TREC-10 were used. This base set of 1000 questions was further reduced by eliminating most definition questions (i.e., "What is X?") because of the difficulties inherent in deciding if any given answer is correct. The correctness of an answer for a definition question nearly always depends on contextual clues which are not present in the TREC context of a list of isolated questions. For example, one unspecified dimension is size. Is the user looking for a single word hypernym, a few sentences, or a 50-page dissertation? Removing most definitional

questions left 822 questions in the list.

These 822 questions were randomly distributed into two 411-question sets, and one set was arbitrarily called the training set, with the other set being the test set. The training set was not used to perform any kind of machine learning, but was used to guide the manual creation of the linguistic analysis algorithm described in Section 4.1. Every part of this algorithm, from the list of question types to the corrections needed to fix the part-of-speech tagger, was developed solely using the training set.

6.2 Evaluation Methodology

For each question in the list of 822, an ordered list of candidate answers as generated by Aranea was provided, along with supporting information for each of the candidates. This supporting information was a list of text snippets returned from a web search that Aranea judged to be relevant to the particular answer candidate.

For the baseline Aranea evaluation, the top-ranked candidate answer was chosen as the official answer for each question, and these official answers were compared to answer pattern files. The pattern files compared against were slightly modified versions of the answer pattern files provided by NIST for the TREC-9 and TREC-10 QA tracks. The primary difference in the files used for this evaluation was to add additional correct answers, since this evaluation did not consider the task of projecting answers into the AQUAINT corpus. For example, the question "Who was the first governor of Alaska?" does not have any answers in the AQUAINT corpus, while in fact there are four correct answers: the first elected governor of the state of Alaska was William A. Egan, the first governor of Russian Alaska was Aleksandr Baranov, the first governor of the Department of Alaska was Brevet Major General Jefferson C. Davis, and the first governor of the territory of Alaska was Walter S. Clark.

To agree with the latest TREC QA track evaluation guidance, only a single answer per question was allowed for this evaluation. In addition, to be judged correct the answer had to match the answer pattern with no extraneous information present.

6.3 Results on Question Sets

Table 6.1 shows the results of running the baseline Aranea system and the VOCAULA-enhanced system against the training and test question sets described above. The first column is the number of questions in each 411-question set for which Aranea's top-ranked answer is correct. The next two columns are the number of questions for which the top-ranked answer is correct with VOCAULA but incorrect without it, and for which the answer is incorrect with VOCAULA but correct without it. The net change shows the overall improvement in the number of questions answered correctly when VOCAULA is part of the system. These results will be discussed in the following chapter.

	Base Aranea	**VOCAULA-Enhanced Aranea**		
		Better	Worse	Net Change
Training Set:	68	15	1	14 (20.6%)
Test Set:	63	8	3	5 (7.9%)

Table 6.1: Question answering performance for Aranea and VOCAULA

Chapter 7

Discussion

7.1 Successes

To analyze the successes and failures of the current VOCAULA implementation, this section focuses on the questions in the training and test sets for which the outcome with VOCAULA was different from the outcome with just the baseline Aranea system. First we look at some examples where the VOCAULA-enhanced system improved upon the baseline.

VOCAULA is quite successful at finding an instantiated inversion in web snippets while ignoring extraneous text. For example, the question "Name one of the major gods of Hinduism? [sic]" and candidate answer "Vishnu" yield the instantiated inversion "Vishnu is one of the major gods of Hinduism" (other inversions are generated, but this is the one of interest in this case). The particular snippet that VOCAULA finds a match in is "... earth Lord Vishnu is one of the three major gods of Hinduism and is called The_Preserver of life Krishna embodies him on earth ...."[1] As you can see, this system is able to ignore extra text before and after the matching phrase, and even ignores extra text within the matching phrase (i.e., "three").

VOCAULA is able to match with even more flexibility than the last example shows. The question "What do you call a group of geese?" and candidate answer "gaggle"

[1]The underscore in The_Preserver is a result of collapsing quoted phrases as described in Section 4.1.1.

yield the instantiated inversion "You call a group of geese a gaggle," but VOCAULA is still able to find a match in the snippet "... a group of geese on the ground is called a gaggle A group of geese in the air is called a skein ...."

VOCAULA also finds answers that, while correct, were probably not anticipated when the question was asked. The question "Who created 'The Muppets'?" generates the inversion "[xx] created The_Muppets," which matches against the snippet "... with his brother Brad as co writer Gilchrist created The_Muppets comic strip for Henson ..." with the candidate answer "Gilchrist." Most people would say that Jim Henson created the Muppets, and in general they would be right. However, there was also a comic strip called "The Muppets" which was based on Henson's Muppets and was created by Guy Gilchrist, so "Gilchrist" is a correct answer to this question. As discussed in Section 6.2, this answer had to be added to the answer pattern file for the evaluation, since the original patterns were written for answers in the AQUAINT corpus and did not anticipate other correct answers. Since the answer "Gilchrist" is not found in the AQUAINT corpus, the Answer Projection module in Aranea (see Figure 3-1) would have to continue to the next candidate answer and try to project it into the corpus.

VOCAULA also matches against text segments in snippets that are not grammatical sentences. This is important because of the large variation in formatting and grammaticality of text on the Web. For example, the question "What is the zip code for Fremont, CA?" and candidate answer "94536" are successfully matched against the snippet "... ZIP Codes Details for FREMONT CA 94536 ...."

VOCAULA is also able to match definitional questions to answers contained in an appositive. The question "What are the colors of the German flag?" matches with the answer "black red and gold" in the snippet "... Its colors are the colors of the German flag Black red and gold ...." The relationship between the main noun and the appositive can also be switched, as in the question "What is the rainiest place on earth?" and candidate answer "Mount Waialeale," which match against the snippet "... Mount Waialeale the rainiest place on earth ...."

The final type of matching VOCAULA is especially successful at is location ques-

tions. Many locations are given as ⟨smaller thing⟩, ⟨larger thing⟩ (e.g., Niagara Falls, New York). VOCAULA takes advantage of this to match questions like "Where is the Kalahari Desert?" to a place-name answer such as "Botswana" (e.g., the snippet "... Kalahari Desert Botswana ..." matches for this question and answer).

7.2 Failures

Despite these successes (and partially because of them) VOCAULA also finds some matches with incorrect answers. When it then promotes an incorrect answer over what used to be a correct answer in the top spot of the baseline Aranea system, the performance of the VOCAULA-enhanced system is degraded. For example, the question "What is the longest suspension bridge in the U.S.?" and candidate answer "1964" match against the snippet "... in 1964 It is the longest suspension bridge in the United States ...." Although syntactically "It" could be referring to "1964," very simple heuristics about valid answers for particular question types could be used to prevent this mistake. Aranea uses some heuristics to filter certain answers (see Section 3.2.3), and these should be incorporated into and improved upon within VOCAULA.

VOCAULA also fails to detect *intrasentential invalidation*, which includes hypothesizing, negation, and joking. For the question "Who invented the paper clip?," the answer "Majority Leader Trent Lott" is considered a match due to the snippet "... a press release says that Senate Majority Leader Trent Lott invented the paper clip in response to Al Gore saying on CNN that he started the internet ...." It is obvious to the human reader that this answer is not correct, but it is not as easy for a computer system to detect. Even a complete syntactic analysis of this snippet will only tell the system that the fact in question was purported by another entity, "a press release." Even a simple semantic analysis would indicate that press releases are likely to contain true facts. Deeper real-world knowledge would be necessary to realize that Al Gore's "invention" of the Internet is a famous pretension, and comparing Trent Lott's "invention" of the paper clip to that indicates that it is probably not accurate either.

The last problem with the current implementation of VOCAULA which caused some degradation in performance is also one of its strongest features—the ability to match against slightly modified phrases if the key features are present. For example, the question "What is the primary language of the Philippines?" and candidate answer "Tagalog" match against the snippet "... based on Tagalog the primary Indonesian language of the Philippines ...." Unfortunately, the single added adjective "Indonesian" makes the answer "Tagalog" correct for the snippet but incorrect for the original question. The simple and obvious answer for this problem is to make sure the set of key features which are matched against is the correct and complete set. What is not so obvious or simple is how to define this set of features. If an exact match is required between all the adjectives in the question and the snippet, very few snippets will contain a match, which is exactly the problem the ITC-IRST team found when creating DIOGENE [21, 22]. On the other hand, if the match is allowed to happen with too much flexibility incorrect answers will match (as in this example). The current matching performed by VOCAULA seems to be a reasonable compromise, but improving this compromise will be an ongoing research area.

Chapter 8

Future Work

8.1 Increased Parser Coverage

There are several areas within VOCAULA where additional improvements have the possibility of significantly raising performance. The first of these is improvement to the partial parser. Since the questions asked are usually grammatical, it should be possible to exactly parse all of them. One of the difficulties with this goal is overcoming or recovering from errors in the part-of-speech tags that are assigned. Some of this recovery is already performed by VOCAULA, but additional work in this area would be fruitful. Also, the list of question patterns in Table 4.2 could be enlarged and refined. More exact determination of the question pattern would allow VOCAULA to better identify the key features of the question that must be present in any correct answer, such as prepositional phrases that further constrain the entity referenced by the subject or object of a sentence. In addition to these system-level improvements, there are larger natural language processing research topics for which improvement would also help VOCAULA. Two examples of these topics are anaphoric reference resolution and prepositional phrase attachment.

The partial parsing of web snippets is one of the primary areas of ongoing research for this system. Due to the minimal requirements for publishing information the Web, spelling, punctuation, and grammar errors are common. Additionally, the Web is designed as a visual display medium, which means that information which would be

obvious to a human observer is disguised if the text on the page is considered by itself. For example, consider a list of definitions on a web page, where each defined word is actually a fancy image of the word and only the definitions are text. To understand this page a system would not only have to realize that the images are part of the presentation, but would have to perform optical character recognition on each image to determine what it says. In simple cases it is not beyond the state of the art to derive extra-textual information from web pages [19], but using snippets prevents this. All extra-textual information has already been removed from the snippets before they are returned by Google, so trying to recover this information would require reviewing the source web page for each snippet and would unacceptably impact system performance.

8.2 Entity Recognition

There are many types of entities that are represented by complex orthographic structures, but reference a single thing. These entity types include names (e.g., people, places, organizations), dates (and other time references), and titles (e.g., *President of the United States*).

The current VOCAULA system does not have an entity recognition component, and instead parses all entities as simply parts of the sentence (typically as noun phrases). Recognizing entities in questions would be a valuable source of key features to match against in web snippets. Since entities are primarily recognized independent of any part-of-speech assessment, this would also surmount any errors in the tagging of these entities. Another useful feature of a good entity recognition component would be normalization of various orthographic forms for the same concept. For example, "JFK," "John Fitzgerald Kennedy," and "President Kennedy" all refer to the same person, and "21 May 2003," "May 21st, '03," and "5.21.03" all refer to the same date. Normalizing these would allow VOCAULA to match against a wider range of snippets without sacrificing accuracy.

8.3 Tighter System Integration

Although VOCAULA is designed to be modular, giving up some of this modularity for tighter integration with the host system (i.e., Aranea) could improve performance in two specific areas: candidate answer ranking and question type analysis. As part of the Knowledge Mining module, Aranea assigns a confidence score to each candidate answer (see Section 3.2.4). These are not currently used in VOCAULA, since their presence is not guaranteed in any other similar system. Even if they are present in another system, their absolute levels would not bear any relationship to the levels Aranea produces, so additional training of VOCAULA would be necessary. Despite this, it would be useful to incorporate Aranea's scores into VOCAULA to see how much of a difference they make. This would allow answers in which Aranea is highly confident to remain top-ranked, perhaps allowing VOCAULA to overcome problems with intrasentential invalidation as discussed in Section 7.2. It would also allow a more intelligent re-ordering of the candidate answers than the current method of simple putting candidates with matching snippets above those without matches.

Another area where integration with Aranea could boost performance is analyzing question types. The information Aranea develops when determining question types for generating exact queries in the Knowledge Mining module (see Section 3.2.1) is not currently used by VOCAULA. Instead of starting from scratch, VOCAULA could use this information to improve its own question analyzing phase, which should provide better linguistic information for matching against web snippets.

Chapter 9

Conclusion

This thesis has described a system and methodology for verifying the correctness of potential answers generated by a high recall question answering system which uses the Web as a significant data source. The implemented system, VOCAULA, uses partial parsing of the question to discover linguistic constraints that are not apparent when the question is considered as a set of keywords. These constraints are checked against the candidate answers and their supporting information to verify which answers fulfill all constraints.

VOCAULA was applied to the output of a baseline Web-based question answering system, Aranea, and evaluated against a test set of questions from past TREC question answering tracks. Using the latest TREC standard for judging questions correct or not, VOCAULA increased the number of correct questions over the baseline Aranea system by nearly ten percent. This is not a huge improvement, but it does confirm the usefulness of an answer verification component in a question answering system based on high recall.

There are several areas in which this system can be improved. Better parsing of the question, answer, and supporting information will allow for better confirmation of linguistic constraints, and more sophisticated question type identification will improve the understanding of those constraints. Also, better integration with the rest of the question answering system will make additional information available to help guide the answer verification process.

VOCAULA demonstrates that it is possible to build a question answering system that begins with a large enough list of candidate answers to ensure sufficiently high recall, and pares down that list using linguistic analysis. This helps achieve the simultaneous goals of high recall and high precision that any knowledge retrieval system must work toward. I hope the work described here moves us closer to a day when a computer system will be able to emulate the abilities of the fabled Oracle at Delphi.

Bibliography

[1] Ricardo Baeza-Yates and Berthier Ribero-Neto. *Modern Information Retrieval.* Addison-Wesley Longman, 1999.

[2] BBC News. Amazing Tales from Planet Tabloid: Contemporary Office Etiquette, January 2001. At `http://news.bbc.co.uk/1/hi/uk/1113955.stm`.

[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. A simple rule-based part of speech tagger. In *Proceedings of the Third Applied Natural Language Processing Conference (ANLP-92)*, 1992.

[5] 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.

[6] Central Intelligence Agency. The World Factbook, 2002. At `http://www.odci.gov/cia/publications/factbook/`.

[7] Nancy A. Chinchor. Overview of MUC-7. In *Proceedings of the Seventh Message Understanding Conference*, 1998.

[8] Abraham Ittycheriah and Salim Roukos. IBM's statistical question answering system – TREC-11. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, 2002.

[9] Han-Min Jung and Gary Geunbae Lee. Multilingual question answering with high portability on relational databases. In *Proceedings of the Nineteenth International Conference on Computational Linguistics (COLING-2002)*, 2002.

[10] 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.

[11] Boris Katz. The START natural language question answering system, 2003. At `http://www.ai.mit.edu/projects/InfoLab/`.

[12] 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.

[13] Boris Katz, Jimmy Lin, and Sue Felshin. The START multimedia information system: Current technology and future directions. In *Proceedings of the International Workshop on Multimedia Information Systems (MIS 2002)*, 2002.

[14] Boris Katz and Patrick H. Winston. Parsing and generating English using commutative transformations. AI Memo 677, MIT Artificial Intelligence Laboratory, 1982.

[15] K. Kwok, L. Grunfeld, N. Dinstl, and M. Chan. TREC-9 cross-language, Web and question-answering track experiments using PIRCS. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, 2000.

[16] Jimmy Lin. Indexing and retrieving natural language using ternary expressions. Master's thesis, Massachusetts Institute of Technology, 2001.

[17] 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.

[18] Jimmy Lin, Aaron Fernandes, Boris Katz, Gregory Marton, and Stefanie Tellex. Extracting answers from the Web using knowledge annotation and knowledge mining techniques. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, 2002.

[19] D. Lopresti and J. Zhou. Locating and recognizing text in WWW images. *Information Retrieval*, 2:177–206, 2000.

[20] 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.

[21] Bernardo Magnini, Matteo Negri, Roberto Prevete, and Hristo Tanev. Comparing statistical and content-based techniques for answer validation on the web. In *Proceedings of the VIII Convegno AI*IA*, 2002.

[22] Bernardo Magnini, Matteo Negri, Roberto Prevete, and Hristo Tanev. Mining knowledge from repeated co-occurrences: DIOGENE at TREC 2002. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, 2002.

[23] Joe Messinger. They shot for the moon — astronaut photographs by Frank Messinger, 2003. At `http://www.geocities.com/CapeCanaveral/Launchpad/1122/biggene.html`.

[24] Eric Nyberg. Open-domain question answering. Class 15-381 spring lecture, Carnegie Mellon School of Computer Science, 2003. At `http://www.cs.cmu.edu/~15381/Lectures/15381_QA.ppt`.

[25] Eric Nyberg, Teruko Mitamura, J. Carbonell, J. Callan, K. Collins-Thompson, K. Czuba, M. Duggan, L. Hiyakumoto, N. Hu, Y. Huang, J. Ko, L. V. Lita,

S. Murtagh, V. Pedro, and D. Svoboda. The JAVELIN question-answering system at TREC 2002. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, 2002.

[26] *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.* Association for Computing Machinery's Special Interest Group on Information Retrieval, 1973.

[27] Amanda Spink, Detmar Wolfram, B. J. Jansen, and Tefko Saracevic. Searching the Web: The public and their queries. *Journal of the American Society of Information Science and Technology*, February 2001.

[28] Students for the Exploration and Development of Space. At `www.seds.org`.

[29] Students for the Exploration and Development of Space. The Messier database. At `http://seds.lpl.arizona.edu/messier/Messier.html`.

[30] Beth Sundheim. Navy tactical messages: Examples for text-understanding technology. Technical Document 1060, Naval Ocean Systems Center, 1987.

[31] Today in Science History. December 7 - Events. At `http://www.todayinsci.com/12/12_07.htm`.

[32] P. Utgoff, editor. *Proceedings of the TIPSTER Text Program (Phase I).* Morgan Kaufmann, 1988.

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

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

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

[36] Ellen M. Voorhees and Lori P. Buckland, editors. *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002).* National Institute of Standards and Technology, 2002.

[37] Ellen M. Voorhees and Dawn M. Tice. The TREC-8 question answering track evaluation. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, 1999.

[38] 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.

[39] William A. Woods, Ronald M. Kaplan, and Bonnie L. Nash-Webber. The lunar sciences natural lanaugage information system: Final report. BBN Report 2378, Bolt, Beranek and Newman, 1972.

[40] George Kingley Zipf. *The Psycho-Biology of Language.* MIT Press, Cambridge, Massachusetts, 1935.