Answering multiple questions on a topic from heterogeneous resources

Boris Katz, Matthew Bilotti, Sue Felshin, Aaron Fernandes, Wesley Hildebrandt, Roni Katzir, Jimmy Lin, Daniel Loreto, Gregory Marton, Federico Mora, Ozlem Uzuner
MIT Computer Science and Artificial Intelligence Laboratory
Cambridge, MA 02139

1 Introduction

MIT CSAIL's entry into this year's TREC Question Answering track focused on the conversational aspect of this year's task, on improving the coverage of our list and definition systems, and on an infrastructure to generalize our TREC-specific tools for other question answering tasks.

While our overall architecture remained largely unchanged from last year, we have built on our strengths for each component: our web-based factoid engine was adapted for input from a new web search engine; our list engine's knowledge base expanded from 150 to over 3000 lists; our definitional nugget extractor now has expanded and improved patterns with improved component precision and recall.

Beyond their internal improvements, these components were adapted to a larger conversational framework that passed information about the topic[1] to factoids and lists. Answer selection for definitional[2] questions newly took into account the prior questions and answers for duplicate removal.

Our factoid engine, Aranea (Lin et al., 2002; Katz et al., 2003), used the World Wide Web to find candidate answers to the given question, and then projects its best candidates onto the corpus, choosing the one best supported. This year, instead of using only Google for web search, we integrated results from the Teoma search engine as well.

Our list engine, Pauchok (Tellex et al., 2003), retrieved passage-sized chunks of text relevant to the question using information retrieval techniques, and projected onto them the fixed lists associated with the question focus. This year we used several new techniques and knowledge sources to gather many times more fixed lists than we had last year.

Our definition engine, Col. ForBIN (Hildebrandt et al., 2004; Fernandes, 2004), inspects the text collection for syntactic patterns often associated with a definitional context, and extracts pairs of targets and definitional nuggets. Topics are then matched against a database of target–nugget pairs. We have expanded the number of patterns and their complexity, yielding improved extraction performance, but changed target matching in a way that caused a net loss in accuracy. A new anaphor-resolution engine improved our final score.

We have made several infrastructure improvements to the original AQUAINT data set: we made it XML-compliant, separated conjoined articles, extracted metadata, and removed meta-text. We also created a standoff XML annotation architecture for storing intermediate processing stages (e.g., POS tags), that was used by both the list and definition engines.

Many unforseen technical challenges forced us to cut integration and testing short, so that many new features were never compared to old ones. This caused, in some cases, no answers, or remarkably poor answers, which were easily fixed after the fact.

We will expand on each of these topics:

2 Question Analysis

The test collection contained 65 "targets", which in this paper we will call "topics" to dif-

[1] "target" in the Guidelines' terms. See Section 2.
[2] "other" in the Guidelines' terms.

ferentiate them from the "focus" (which has been called the "target" of a factoid or list question) of each individual question within the topic. The primary tool we used for analyzing each question in the context of its topic was the START Natural Language Question Answering System[3] (Katz et al., 2002; Katz, 1988; Katz, 1997). Three of its internal functions were exposed in a TREC-specific API, and enhanced to work with a wider array of questions:

- Noun-phrase parsing for the topic itself,
- anaphoric substitution to place the topic into each question as appropriate, and
- focus extraction to find for each question the type of answer sought.

For example, START would analyze "boxer Floyd Patterson" as an occupation and a person, choosing to substitute only the name into a question: "How old was Floyd Patterson when Floyd Patterson won the title?", or "List the names of boxers Floyd Patterson fought." The algorithm is shown in Figure 1.

In the case above, the factoid question would get passed to Aranea with the occupation "boxer" appended: Aranea analyzes only the beginning of the question to find the expected answer-type, and uses just keywords thereafter.

The list question is more closely coupled, and our list engine Pauchok was told via the API that "boxer(s)" was the focus of the question, and separately that Floyd Patterson's occupation was boxer.

The definition processor was given the topic unanalyzed.

If the query-analysis algorithm failed to find a substitution for the topic into a query, then both the factoid and list engines simply appended the topic to the query for document retrieval.

3 Factoid Questions

We have been using the Aranea system for question answering for three years, and were able to deploy it in our updated architecture with few changes. Where it used to send the query to Google[4], it now sends it to Teoma[5] as well, and makes no distinction between the two sources in further processing.

4 List Questions

We retrieved passage-sized chunks of text using information retrieval techniques, and projected fixed lists, whose annotations matched the question focus, onto the passages. The most significant change this year was our accumulation of 20 times the number of fixed lists used last year. The lists were compiled from several sources and provided the backbone of the list question answering mechanism. Examples of the lists extracted are given in Figure 2.

4.1 Focus Identification

The first step for answering list questions is to identify the focus of the question, the "target" in previous years' terminology, which is indicative of the expected answer type. After START has incorporated the topic into the question,[6] it also identifies several candidate structures as possible focus candidates. It provides these to the list engine in the form of a list of strings ordered by specificity.[7]

For example the list for "Name famous people who have been Rhodes scholars" contained:

- "famous people who have been Rhodes scholars"
- "famous people"
- "Rhodes scholars"
- "people"

4.2 Noun-phrase Annotations

Second, each candidate may be associated with several fixed lists, compiled *a priori* by matching noun-phrase annotations for each list. Each of our 3301 separate lists has at least one associated noun-phrase annotation

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

[4] http://www.google.com/

[5] http://www.teoma.com/

[6] or failed to incorporate it and provided it separately

[7] It also provides the parsed data structure, but Pauchok currently uses the list of strings.

1. Find generalizations of the topic
   - either from the structure of the topic
     e.g., "Hale-Bopp comet" is a "comet"; "senator Jim Inhofe" is a "senator",
   - or from pre-existing knowledge of the name
     e.g., a "boll weevil" is a beetle, an insect, an arthropod, ...
2. Make one of the following substitutions, or return failure:
   - any pronoun, respecting gender if available, but not number
     e.g., substitute "Hale-Bopp comet" for "it" in "How often does it approach the earth?"
   - partial topic for whole topic, preserving possessive
     e.g., substitute full topic "Fred Durst" for "Durst" in "Where was Durst born?"
   - topic for generalization
     e.g., substitute "the Berkman Center for Internet and Society", for "the center" in "Where is the center located?"

Figure 1: START's algorithm for query analysis.

that identifies the list, and is matched against the focus identified above. Continuing with the example above, we do not have a list of *famous Rhodes scholars*, nor of *Rhodes scholars*, but we do have lists of *famous people* (78000+ from START's preexisting biography.com knowledge source) and of *people* (using heuristic name matching).

If a list matched a focus, then elements of that list that appear in the retrieved passages are scored based on the rank of their passage.[8] Items from each focus backoff are strictly preferred to items in later backoffs.

The fixed and dynamic lists that this method relies on are described in Sections 4.3 and 4.4.

4.3 Expanded Fixed Lists

Last year we used about 150 manually-compiled lists in a similar list-answering process. This year we used nearly 3300 fixed lists. We semi-automatically extracted lists in three ways: we found hyponyms of words appearing in "Which *X*" context; we found common descriptions of people in first sentences of WorldBook Encyclopedia articles; and we used new semi-structured online resources to compile further categories.

[8]This implies scoring based on document query backoff, because sets of retrieved document chunks for each expanded query are appended to one another.

Our process for "wrapping" semi-structured online resources is well described in papers about our Omnibase system (Katz et al., 2002). This process contributed 171 of our lists.

The WorldBook Encyclopedia's first sentences often contain very salient descriptions, especially for people, that serve as category names. For example from the entry for MacDowell: "MacDowell, Edward Alexander, was an American composer and pianist." we can put him into three categories of famous people: "American composer", "composer", and "pianist". We used a context-free grammar to parse all first-sentences for people, and with some manual cleanup generated 730 lists.

From the corpus itself, we selected category names by looking for "Which *X*" and "What *X*" surface patterns at the beginnings of sentences. We associated these category names with instances of their immediate WordNet hyponyms that appeared in the corpus.

This process generated 11,000 lists, from which we manually selected 2360, based on a subjective coherence of the list elements, subjective quality of the list name as a description of the elements, and having more than one proper-noun list element. During this manual process, we also added synonymous noun-phrase annotations, other than the list name,

Italian region	**cellular phone manufacturers, cell phone manufacturers**	**labor leader, leader**
Abruzzi Basilicata, Lucania Calabria Campania Emilia-Romagna ...	Nokia Samsung Motorola Nextel Sprint	Sidney Hillman Leonard Woodcock Samuel Gompers Elizabeth Gurley Flynn James Riddle Hoffa David Dubinsky
from corpus and WordNet	*from internet sources*	*from World Book*

Figure 2: Examples of lists extracted from the corpus. The noun-phrase annotation is shown in bold. Multiple entries on a line indicate "synonyms", of which only one form will ever be reported, even if both are found.

that could be used to ask about the list.

The final set of 3091 automatically extracted lists contain a total of 30,112 symbols.

4.4 Dynamic Lists

Some "lists" are actually scripts. For example the annotation that matches books by an author goes to Barnes-and-Noble.com to identify the author's works, and returns these works as the "fixed" list to work from.

In addition, if we did not have a noun-phrase annotation match for any of the focus backoffs, but any of the backoffs were in WordNet, then we treated its WordNet hyponyms as if they were a known fixed list.

4.5 Guess Answers

Despite a far greater number of fixed lists than last year, we cannot anticipate every category a question might be asked about. When we do not match any annotation for a focus, we use heuristics to find reasonable candidates, called "guess answers".

A guess answer can be a noun phrase or a quoted title. To be selected, the guess answer must appear within a short fixed linear distance of a focus string.

For example, if we are asked for kinds of grapes, provided we have no list of grape types, we prefer the noun phrases "chardonnay grapes" and "grape juice" over "grapes like pinot and chardonnay", and those over "grapes of distinction like pinot and chardonnay", where the grape types receive no score.[9]

Our two runs varied how guess answers were used. In our first run, guess answers were provided for every question. In our second run, guess answers were used only when no fixed lists were identified to match the question focus.

4.6 Passage Retrieval

Passages to project lists onto were retrieved with a much simpler algorithm than last year: we indexed groups of paragraphs that were at least 500 characters long as documents for IR, and then used document retrieval technologies described in (Bilotti et al., 2004) to retrieve these chunks, treating them as passages.

We did a preliminary investigation to see how much worse this method was than the method we used last year, and found anecdotally that it was comparable. Because of the difficulty we anticipated in adapting last year's passage retrieval code to the changed query expansion and document retrieval modules, we chose to focus our efforts elsewhere.

We chose 500 characters as a minimum passage size in order to avoid paragraphs separated for effect, but not containing enough text to give context to its contents.

[9]Of course the noun phrase "grapes of distinction" *would* be incorrectly collected as a guess answer.

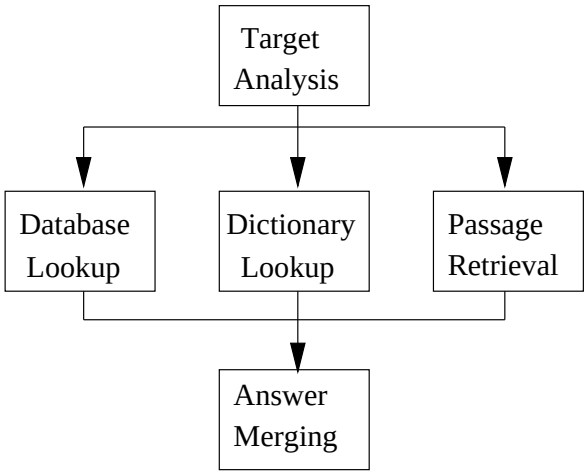

Figure 3: Architecture for answering definition questions.

4.7 Answer Selection

Answer selection was based on duplicate removal and IR-based scoring. Two answers were considered duplicates if all of their content words (stopwords and punctuation removed, case ignored) were the same. If one set of content words subsumed the other, the longer answer was chosen. In both of our runs, the top 25% of guess answers were to be returned if START did not identify a focus. In our mit1 run, that is the only time that guess answers were returned. In our mit2 run, the top 25% of guess answers were to be returned for every question.

5 Definition Questions

Our architecture for answering definitional questions is shown in Figure 3. Unlike last year, the topic to define is provided. That topic is then passed to three parallel techniques for finding definitional "nuggets" of the topic: lookup in a database of relational information created from the AQUAINT corpus, lookup in a Web dictionary followed by answer projection, and lookup directly in the AQUAINT corpus with information retrieval techniques. Answers from the three different sources are merged to produce the final system output. The following subsections briefly describe each of these techniques.

5.1 Target Analysis

This year the targets to define are not couched in a question, but they may still not be in a form that would likely be found in the corpus or in a dictionary. For example "the band" in "the band Nirvana" serves to disambiguate the sense of "Nirvana", but mentions of that band will more frequently appear as just "Nirvana" than as the whole phrase. Looking for "Nirvana" alone, we still must find instances that refer to the band.

5.2 Database Lookup

The use of surface patterns for answer extraction has proven to be an effective strategy for question answering. We began last year to use surface patterns to extract a database of definitional nuggets from the corpus from which to later answer questions, and have expanded on the sophistication and variety of those surface patterns for this year. We look for definitional patterns *a priori* to overcome a preponderance of non-definitional contexts for target words in document retrieval results; compile-time processing gives us better recall. We have reimplemented last year's system, and given it the name Col. ForBIN.[10]

Last year we had copular, appositive, and occupation patterns, and a few verb patterns ("became", "founded", "invented", etc.). (Hildebrandt et al., 2004) This year we have sixteen classes of patterns. They are comprised of cascades of regular expression patterns, that capture among other things: base noun phrases, single-level, two-level, and recursive noun phrases, prepositional phrases, relative clauses, and tensed verbs with modals. The new patterns allowed us to identify the target and nugget as any constituents of the matched pattern, so we were able to focus on finding exact definitional nuggets rather than windows of definitional contexts.

Finally, we have incorporated BBN's IdentiFinder program (Bikel et al., 1999) to iden-

[10]Phish's song *Col. Forbin's Ascent* claims: "Col. Forbin, I know why you've come here, / And I'll help you with the quest to gain the knowledge that you lack."

tify named entities. Most of the patterns require that their target entity be an IdentiFound named entity.

5.3 Improvement in Patterns

To measure precision of the pattern extraction engine, we asked students to annotate at least 200 sentences marked by the pattern extractor for each pattern. To measure recall, we asked students to annotate randomly selected sentences from documents judged relevant to TREC12 definition questions. See Figure 4 for component results.

In each case, they were asked to mark target–nugget pairs, where the nugget was a good description of the target. Because the patterns between the two versions are different, the precision judgements on each pattern were not comparable, so we effectively had different test sets for TREC12 precision and Col. ForBIN precision measurements. Another problem comparing precision was that we used data from the TREC12 judgements, and possibly[11] some of the data from the newer pattern judgements, in the evaluation. The recall judgements are the same for both cases, and were unseen by developers.

Much of the improved precision came from restricting targets to named entities, and from ensuring that those targets were the thing described rather than being simply linearly adjacent. Much of the improved recall came from the new ability to associate multiple nuggets, and non-adjacent nuggets, with a target.

Examples of sentences from the corpus matching each pattern are shown in Figure 5, with emphasis on targets from this year's competition. The composition of the patterns, the testing methodology, and the results, are detailed in (Fernandes, 2004).

5.4 Referring Expression Resolution

We used a simple rule-based referring expression resolution engine to assign full names to definite noun phrases, partial names, and pronouns in the entire corpus using an algorithm very similar to that used in query analysis. Referring expressions had to match their referent's gender and number where available, and preference was given to referents based on linear distance to their latest mention.

As in query analysis, partial names were expanded to their full known name, but without any further known description (e.g., "Floyd Patterson" would be substituted for "Patterson", sans "boxer". Unlike query analysis, no ontological information was used: only simple definite noun phrase references and occupation references were expanded.

Component performance was evaluated against the MUC-7 data set, yielding 71% precision and 23% recall.

5.5 Dictionary Lookup

Another component of our system for answering definitional questions utilizes an existing Web-based dictionary for nuggets. This component is largely unchanged from last year. Obviously, such an approach cannot be applied directly, because all nuggets must originate from the AQUAINT corpus. So we use answer projection techniques to "map" dictionary definitions onto AQUAINT documents.

Given the topic, our dictionary lookup engine goes to the Merriam-Webster online dictionary for its definitions. Keywords from the definition are used in a Lucene query to retrieve documents, and to score sentences based on keyword overlap with the dictionary definition. Sentences are trimmed to 250 bytes around the topic, containing the beginning or end of sentence if possible.

5.6 Document Lookup

Finally, our system looks for answers in the AQUAINT corpus itself, using the topic as a Lucene query, and selecting sentences that contain the topic. As in dictionary lookup, these sentences are trimmed to 250 bytes around the topic, containing the beginning or end of sentence if possible.

[11] The students doing annotation and development worked together; we were not sufficiently careful to keep the tasks separate.

Version	Recall		Precision	
	exact	inexact	exact	inexact
TREC 12	144 / 483 = .30	175 / 483 = .36	1410/7061 = .19	3114/7061 = .44
Col. ForBIN	156 / 483 = .32	186 / 483 = .39	2527/4190 = .60	2669/4190 = .64

Figure 4: Comparison of definition extraction component from TREC 12 to present. Precision is evaluated on separate but comparable data sets. Recall is measured on one set of data from articles judged relevant to TREC12 definition questions.

5.7 Topic Matching and Answer Selection

Like last year, we looked for answers in parallel in the database described above, from a dictionary source, and from the corpus itself. Unlike last year, we did not prefer database answers strictly over dictionary answers, and those in turn over plain corpus answers. Rather, we weighted candidate sentences from the database at three times their score, and from dictionary at twice their score, and let them mix.

The answers were then presented by target quality. All answers matching a better target were presented before any matching a worse one. Unfortunately we were unable to use START's backoff mechanism to identify the relevant portion of the topic (e.g., "Floyd Patterson", then "Patterson" for "boxer Floyd Patterson"). Instead, we used a combination of candidate target precision and recall for quality. This made no distinction between "Fred" and "Durst" as backoffs for the singer.

Within each target, answers were ranked by novelty—the amount of word overlap between that answer and any previous answer. The base score of each word was its *idf* in the corpus, but this was boosted if the answer appeared often in the candidate matches yet to be printed (as a measure of salience), or if it was all lower-case (to promote answers with more English text in them over lists of names). The maximally novel answer was selected at each step, and novelties recalculated with that answer now among the set of previous answers.

The novelty scoring was initialized with the previous questions and answers to avoid duplication of answers already given.

6 Infrastructure Improvements

One of the most challenging components of any question answering endeavor is the complexity of the input data. When the complexity is in language, the challenge is welcome and exciting. Complexity in the input format is simply frustrating. We made a strong effort this year to clean the AQUAINT corpus, making the following changes:

- transform character entities and tags from SGML to XML
- separate the title into its own tag
- separate any article abstract into its own tag
- separate metadata such as:
  - Reporter
  - Location
  - Source
- remove comments to/from editors
- separate documents that contain summaries on multiple news stories into a document for each story
- remove duplicate documents (leaving pointers to the documents they duplicate)

We plan to make the scripts for cleaning and access available to other TREC participants. We hope that this common cleaned corpus will lower one barrier to entry into the competition.

Another infrastructure improvement we made was to create a suite of standoff XML tools in perl and java to manipulate, serialize, and display XML annotations in the text. This is undergoing revisions from lessons learned, but we also hope to make this code available.

pattern	example
copular:	**Ray Rhodes** was *coach of the year.* APW19990102.0072
copular w/anaphora:	**He** [**Franz Kafka**] was *one of the best-known Czech authors of early 20th century.* APW20000425.0204
affiliation:	When a note detailing the idea reached *GE* chairman **Jack Welch**, NYT19991021.0177
occupation:	When a note detailing the idea reached GE *chairman* **Jack Welch**, NYT19991021.0177
occupation:	*Singer-choreographer* **Fred Durst** wants a multimedia empire... NYT19990705.0170
age:	**Adams**, *30*, a convicted murderer, was fatally shot in March 1994 while fighting with another prisoner. APW19990101.0028
appositive:	**Adams**, 30, *a convicted murderer*, was fatally shot in March 1994 while fighting with another prisoner. APW19990101.0028
entity in appositive:	The *disease*, **sporadic fatal insomnia**, is caused by ... APW19990526.0110
also known as:	...caused by the same type of *deformed proteins*, known as **prions**, that ... APW19990526.0110
also called:	Some women in Beijing have established *a non-governmental organization* called **Global Village** to increase awareness of environmental protection.
named:	In the early hours of June 8, 1924, *a 38-year-old British schoolteacher* named **George Mallory** set forth... NYT19990504.0349
like:	Jiang has shown every sign that he aspires to enter the pantheon of *great Communist philosopher-leaders* like **Mao** and **Deng**. NYT19990503.0106
like (false positive):	But although he's *a high-wire act onstage*, like **Iggy Pop**, **Durst** comes across as mellow offstage. NYT19990618.0182
such as:	In the past, researchers had tested various single *nutrients*, such as **calcium**, **magnesium** and **potassium**, to find clues about what affects blood pressure,
became:	**Jennifer Capriati** *became the youngest Grand Slam semifinalist and beat five top-10 players in her first year.* APW19990108.0333
was named:	A year later **he** [**George W. Bates**] was named *managing editor of the International Herald Tribune in Paris.* APW19990107.0283
relative clause:	**Dean**, who *died at age 24 in a 1955 car crash*, is ...
verb-passive:	**Franz Kafka** *was born in Prague, Czechoslovakia, in 1883 and died a month before his 41st birthday, having long suffered from tuberculosis.* APW20000223.0092
verb-pp:	**Mrs. Dole** *served as transportation secretary for President Reagan and labor secretary for President Bush.* APW19990105.0044
verb-np-generic:	**Harding**, who denied advance knowledge, *received probation after pleading guilty to conspiracy to hinder prosecution.* APW19990105.0220

Figure 5: Sample nuggets extracted from the AQUAINT corpus using surface patterns. The **target terms** are in bold, the *nuggets* are in italics.

	mit1	mit2
Factoid	Aranea was used unmodified for both runs.	
List	always guess	guess unless known list
Definition	without reference resolution	with reference resolution

Figure 6: Differences between the mit1 and mit2 runs.

	Factoid	**List**	**Other**	**Final**
best	.770	.622	.460	
mit1	.313	.119	.184	.232
mit2	.313	.113	.186	.231
median	.170	.094	.184	

Figure 7: Overall system performance

We will be able to make available our standoff annotations for, e.g., Brill tags over the corpus (62Gb). We have found it much faster to read such tags from a file than to regenerate them on the fly.

7 Results

We submitted two runs, summarized in Figure 6, in which we tested the effect of referring expression resolution on definition questions, and the effect of using or not using the best guess answers returned by our list component.

The referring expression resolution component (Section 5.4) improved recall for five definition questions and lowered recall for four. The paired difference between F-measures of the runs was .0029 $\pm$ .0153 (p-value: .353) and so was not statistically significant.

The use of guess answers (Section 4.7) improved both precision and recall for two questions, 62.4 and 63.3, but the paired difference between the two runs was still not significant: 0.0058 $\pm$ 0.0082; p-value 0.0798.

In hindsight, we believe that some promising ideas were overshadowed by mistakes in our software engineering process, primarily in insufficient integration and testing, and we look forward to fielding a more robust entry in the next competition.

References

Daniel M. Bikel, Richard L. Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34(1-3):211–231.

Matthew Bilotti, Boris Katz, and Jimmy Lin. 2004. What works better for question answering: Stemming or morphological query expansion? In *Proceedings of the SIGIR 2004 Workshop IR4QA: Information Retrieval for Question Answering*, July.

Aaron Fernandes. 2004. Answering definitional questions before they are asked. Master's thesis, Massachusetts Institute of Technology.

Wesley Hildebrandt, Boris Katz, and Jimmy Lin. 2004. Answering definition questions with multiple knowledge sources. In *Proceedings of the 2004 Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics Annual Meeting, HLT/NAACL-04*, April.

Boris Katz, Sue Felshin, Deniz Yuret, Ali Ibrahim, Jimmy Lin, Gregory Marton, Alton Jerome McFarland, and Baris Temelkuran. 2002. 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)*, June.

Boris Katz, Jimmy Lin, Daniel Loreto, Wesley Hildebrandt, Matthew Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, and Federico Mora. 2003. Integrating web-based and corpus-based techniques for question answering. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, November.

Boris Katz. 1988. Using English for indexing and retrieving. In *Proceedings of the 1st RIAO Conference on User-Oriented Content-Based Text and Image Handling (RIAO '88)*.

Boris Katz. 1997. Annotating the world wide web using natural language. In *Proceedings of the Conference on the Computer-Assisted Searching on the Internet, RIAO97*.

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

Stefanie Tellex, Boris Katz, Jimmy Lin, Gregory Marton, and Aaron Fernandes. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, July.