Actions, Answers, and Uncertainty: A Decision-Making Perspective on Web-Based Question Answering

David Azari
Computer Science & Engineering
University of Washington
Seattle, WA 98195-2350
azari@cs.washington.edu
tel: 206/543-1695
fax: 206/543-2969

Eric Horvitz
Microsoft Research
One Microsoft Way
Redmond, WA 98033
horvitz@microsoft.com
tel: 425/706-2127
fax: 425/936-7329

Susan Dumais
Microsoft Research
One Microsoft Way
Redmond, WA 98033
sdumais@microsoft.com
tel: 425/706-8049
fax: 425/936-7329

Eric Brill
Microsoft Research
One Microsoft Way
Redmond, WA 98033
brill@microsoft.com
tel: 425/706-4992
fax: 425/936-7329

[†] Abstract

We present research on methods for generating answers to freely posed questions, based upon information drawn from the Web. The methods exploit the typical redundancy of information on the Web by making multiple queries to search engines and then combining the search results into an answer. We focus on the pursuit of techniques for guiding information gathering in support of answering questions via the learning of probabilistic models that predict the value of information drawn from the Web. We first review research on question-answering systems. Then, we present AskMSR, a prototype Web-based question-answering system. We describe the learning of Bayesian-network models that predict the likelihood that answers are correct, based on multiple observations. We review a two-phased Bayesian analysis and present an expected-utility analysis of information-gathering policies using these inferences. After reviewing the results of a set of experiments, we describe research directions. [‡]

Keywords

Question answering, Bayesian networks, information retrieval, cost-benefit analysis, decision theory

---

[†] Reprint requests should be addressed to Eric Horvitz (horvitz@microsoft.com), Microsoft Research, One Microsoft Way, Redmond, WA 98033.

[‡] This paper is an extended version of the conference article, D. Azari, E. Horvitz, S. Dumais, E. Brill. Web-based question answering: A decision making perspective. *Proceedings of the Conference on Uncertainty and Artificial Intelligence*, 2003, pp. 11-19.

1 Introduction

Computer scientists have long pursued the goal of developing computational machinery with the ability to generate answers to freely-posed questions. General question-answering systems depend on techniques for analyzing questions and for composing answers from some corpus of knowledge. This is a challenging problem because the corpus may not contain an explicit matching answer or may contain multiple variants of relevant answers or answer components.

We have been studying procedures that enlist the poorly-structured but copious resources of the Web for answering questions. We seek to exploit the redundancy of information on the Web by making multiple queries to Web search engines and integrating search results into an answer. A key challenge in harnessing the copious, yet ill-structured content of the web is characterizing the value associated with different kinds of informational probes used by a question answering system.

One approach to constructing answers relies on procedures for converting questions into sets of words that may appear in the source corpora. A variety of *query-rewriting* procedures are applied to convert questions into sets of queries posed to search engines, and techniques are employed to convert one or more query results into an answer. To date, there has been little understanding of the value of alternate query rewriting strategies and answer composition methods. We also have little knowledge about the enhancement of the quality of answers with the issuance of increasing numbers of queries to search engines. Given the burden that widely fielded question-answering systems can place on search engines, gaining a deeper understanding of the nature and number of query rewrites is important for deploying real-world question-answering systems.

We describe an investigation of probabilistic modeling and decision analyses to characterize and control querying policies in Web-based question answering. We first provide a brief review of prior work on question answering. We focus particularly on a system developed at Microsoft Research, named AskMSR. We present the rewrite procedures and answer composition methods performed by AskMSR. Then, we review the learning of Bayesian graphical models which can predict the value of alternate rewrite strategies for generating answers to questions. Beyond exploring alternate rewrite procedures, we also studied the influence of the quantity of rewrites on the quality of answers. Such an analysis relies on developing a strategy for ordering queries by their expected value, so as to allow the learning of models that can reason about the costs and benefits of employing additional numbers of queries to the web. We describe the methods we developed and review a set of experiments that demonstrate the effectiveness of the cost-benefit procedures. Although we focus our attention in this paper on methods for guiding the allocation of search resources in a Web-based question-answering system, we believe that the overall methodology of coupling machine learning with decision making under uncertainty provides a promising basis for addressing a spectrum of information-retrieval problems that grapple with tradeoffs in cost and quality.

2 Question Answering Systems

Most text-retrieval systems operate at the level of entire documents. For example, in searching the web, pointers to complete web pages or documents are returned. There has been a recent surge of interest in finer-grained analyses focused on methods for obtaining *answers* to *questions* rather than retrieving potentially relevant documents or best-matching passages from queries—tasks information retrieval (IR) systems typically perform. Approaches to question answering have relied on the application of several key concepts from information retrieval, information extraction, machine learning, and natural language processing (NLP).

Automatic question answering from a single, constrained corpus is extremely challenging. Consider the difficulty of gleaning an answer to the question "*Who killed Abraham Lincoln?*" from a source which contains only the text *"John Wilkes Booth altered history with a bullet. He will forever be known as the man who ended Abraham Lincoln's life."* As Brill, Dumais and Banko (2002) have shown, however, question answering is far easier when the vast resources of the Web are brought to bear, since hundreds of Web pages contain the literal string "*killed Abraham Lincoln,*" providing multiple opportunities for matching and composition.

2.1 Approaches to Question Answering

The TREC Question Answering Track (*e.g.,* Voorhees & Harman, 2001) has motivated much of the recent work in the field of question answering. Most efforts in question answering have focused on fact-based, short-answer questions such as "*Who killed Abraham Lincoln?*", "*What was the length of the Wright brothers first flight?", "When did CNN begin broadcasting"* or "*What two US biochemists won the Nobel Prize in medicine in 1992?"*

Question-answering systems have typically used NLP analyses to augment standard information retrieval techniques. Systems often identify candidate passages using IR techniques, and then perform more detailed linguistic analyses of both the question and matching passages to find specific answers. A variety of linguistic resources (part-of-speech tagging, parsing, named entity extraction, semantic relations, dictionaries, etc.) are used to support question answering. The

Falcon system by Harabagiu, Moldovan, Pasca, Mihalcea, Surdeanu, Bunescu, Girju, Rus and Morarescu (2001) is typical of the linguistic approaches and has demonstrated excellent performance in benchmark tests. In the system, a query is parsed to identify important entities and to suggest a likely answer type. A rich taxonomy of answer types has been developed using lexico-semantic resources from WordNet (Miller, 1995). WordNet represents more than 100,000 English nouns, verbs, adjectives and adverbs composed into conceptual synonym sets and relationships among them, as encoded by lexicographers over the course of many years. Candidate matching paragraphs are similarly analyzed to see if they match the expected answer type. Often, relevant passages will not share words with the query. In these cases, the Falcon system uses WordNet to examine morphological alternatives, lexical alternatives (*e.g.*, nouns "killer," "assassin," or "slayer" will match the verb "killed"), and semantic alternatives (*e.g.*, "cause the death of"). Additional abductive processes are also used to provide answer justification and rule out erroneous answers.

2.2 Web Question Answering

In contrast to these rich natural language approaches, others have developed question answering systems that attempt to solve the difficult matching and extraction problems by leveraging large amounts of data. AskMSR (Brill, Dumais, and Banko., 2002; Dumais, Banko, Brill, Lin & Ng, 2002) is an example of such a system, and one that we explore in more detail in this paper. The main idea behind the AskMSR system is to exploit the redundancy provided by the web to support question answering. Redundancy, as captured by multiple, differently phrased answer occurrences, facilitates question answering in two important ways. First, the larger the information source, the more likely it is that answers bearing close resemblance to the query can be found. It is quite straightforward to identify the answer to "*Who killed Abraham Lincoln?*" given the text, "*John Wilkes Booth killed Abraham Lincoln in Ford's theater.*" Second, even when no exact answer can be found, redundancy can facilitate the recognition of answers by enabling procedures to accumulate evidence across multiple matching passages.

Other researchers have also looked to the web as a resource for question answering. The Mulder system (Kwok, Etzioni & Weld, 2001) is similar to AskMSR in many respects. For each question, Mulder submits multiple queries to a web search engine and analyzes the results. Mulder does sophisticated parsing of the query and the full text of retrieved pages to identify answer candidates. Mulder also employs a local database of term weights for answer extraction and selection. Mulder has not been evaluated with TREC queries, so its performance is difficult to compare with other systems.

Clarke, Cormack & Lyman 2001) investigated the importance of redundancy in their question answering system. They found that the best weighting of passages for question answering involves using both passage frequency (what they refer to as *redundancy*) and a global term weight. They also found that analyzing more top-ranked passages was helpful in some cases but not in others. Their system builds a full-content index of a document collection, in this case the TREC test collection. Their implementation requires that an auxiliary web corpus be available for full-text analysis and global term weighting.

Kwok et al. (2001) and Clarke et al. (2001) perform complex parsing and entity extraction for both queries and best matching web pages, which limits the number of web pages that they can analyze in detail. They also require term weighting for selecting or ranking the best-matching passages which requires auxiliary data structures. AskMSR is distinguished from these in its simplicity and efficiency. The system only uses simple rewrites and string matching, and makes direct use of summaries and simple ranking returned from queries to web resources. The data-driven techniques perform well in TREC benchmark tests (Voorhees & Harman, 2001).

3 AskMSR System

We now turn to reviewing details of the operation of AskMSR as background for our efforts to extend the heuristic system via introducing probabilistic models and automated decision making. After reviewing AskMSR, we will describe our work to develop Bayesian models of the performance of AskMSR components, and to integrate cost-benefit strategies for guiding the system's actions.

The design of AskMSR was motivated by several efforts within NLP research that have demonstrated that, for many applications, significant improvements in accuracy can be attained by significantly increasing the amount of data used for learning. Following the same guiding principle, the tremendous data resources that the Web provides was used as the backbone of AskMSR. The overall pipeline of analysis for AskMSR is displayed in Figure 1.

AskMSR contains two main components, *query rewriting* and *answer composition,* which consists of several sub-processes, including *n*-gram mining, filtering, and tiling (see Brill et al., 2002; Dumais et al., 2002 for details).

3.2 Query Rewriting

AskMSR reformulates each user question into likely substrings of declarative answers to the question. For each question, several rewrites are generated using eight rewrite heuristics. The rewrites vary from specific string matching to a simple "ANDing" of all the query words. As an example, for the query "*Who killed Abraham Lincoln?*" there are three rewrites:

<LEFT> “killed Abraham Lincoln”; “Abraham Lincoln was killed by” <RIGHT>; and killed AND Abraham AND Lincoln. <LEFT> and <RIGHT> refer to the likely placement of candidate answers. The first two rewrites require that a text on the Web match the exact phrase, such as “killed Abraham Lincoln.” We refer to the last rewrite as a conjunctional *back-off strategy*, as it simply "ANDs" together all the query words, leading to less specific queries. The rewrite strings are formulated as search engine queries and sent to a search engine from which page summaries are collected. Any search engine can be used as the provider of results to the second stage of AskMSR’s analysis.

AskMSR assigns heuristic scores to results of different kinds of rewrites. The system assigns higher weights to the results of more precise rewrites than it does to the more general, conjunctional back-off rewrite. This weight assignment has received empirical support in experiments reported by Brill et al. 2002. The decision theoretic approach proposed in this paper provides an overall framework for finer tuning rewrites and their weights for individual queries.

3.3 Answer Composition

Several phases of analysis are employed in AskMSR to identify answers to questions from the results returned by searches with query rewrites as follows:

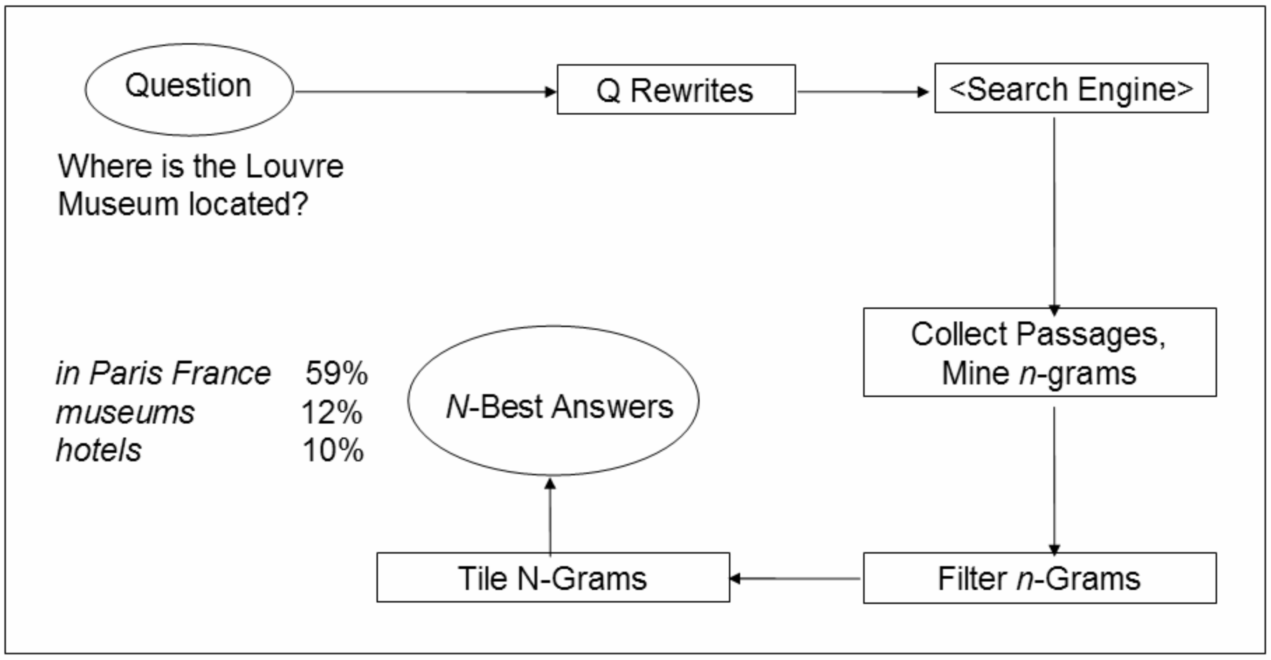


Figure 1. AskMSR analytic pipeline. A set of rewrites are scored and issued to a web search engine, snippets of text returned by the search engine are mined for *n*-grams. The top answers are composed by filtering and tiling the *n*-grams.

Mine N-Grams. From the page summaries returned for each query rewrite, all unigram, bigram and trigram word sequences are extracted. The *n*-grams are scored according to their frequency of occurrence and the weight of the query rewrite that retrieved it. As an example, the common *n*-grams for the example query about the assassination of Abraham Lincoln are: Booth, Wilkes, Wilkes Booth, John Wilkes Booth, bullet, actor, president, Ford's, Gettysburg Address, derringer, assassination, etc.

Filter N-Grams. The *n*-grams are filtered and re-weighted according to how well each candidate matches the expected answer type, as specified by fifteen handwritten filters. These filters use surface-level string features, such as capitalization or the presence of digits. For example, for *When* or *How many* questions, answer strings with numbers are given higher weight, and for *Who* questions, answer strings with capitals are given added weight and those with dates are demoted.

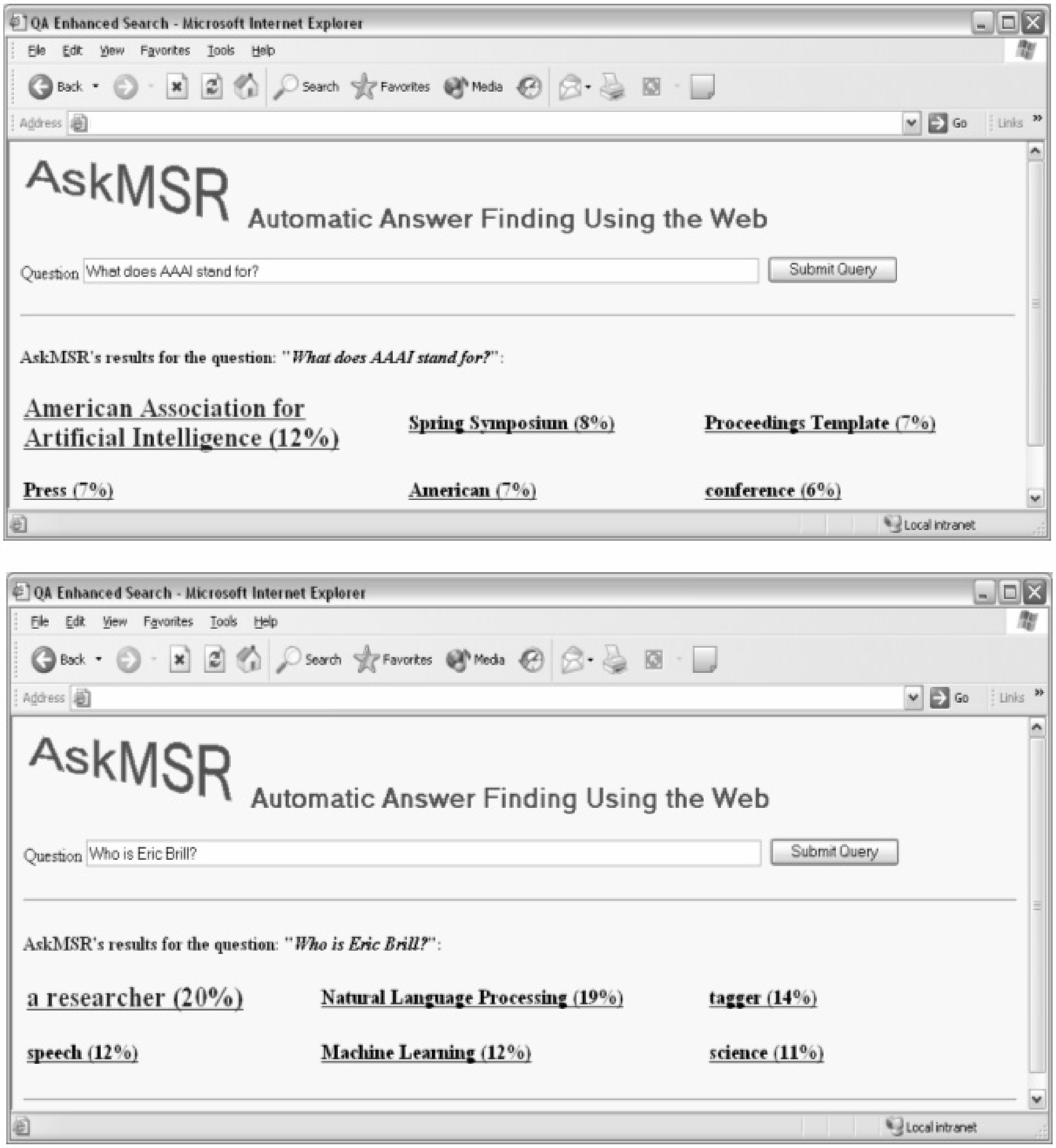
Tile N-Grams. Finally, the *n*-grams are *tiled* together by lining up matching sub-phrases where appropriate, so that longer answers can be assembled from shorter ones. Following tiling, the answers to the example query are: John Wilkes Booth, bullet, president, actor, Ford. John Wilkes Booth receives a much higher score than the other answer candidates because it is found in matches to specific rewrites and because it occurs more often overall.

3.4 AskMSR Prototype

To date, AskMSR has been fielded within Microsoft as a prototype research system. The user interface of a version of AskMSR is displayed in Figure 2. Given a query, the system composes multiple rewrites, mines, filters, and tiles *n*-grams, and then outputs the top answers. Sample queries and top ranked answers for the queries, "*What does AAAI stand for*?" and "*Who is Eric Brill*?" are displayed in Figure 2.

4 Learning about the Value of Queries to the Web

AskMSR's performance has been judged in the TREC question-answering conference to be competitive with the best question answering systems (Voorhees & Harman, 2001). The system can be viewed as deriving its power by employing relatively simple strategies targeted at leveraging the redundancy of the informational content of the Web. Unfortunately, the same mechanisms which provide its power make the system's operation costly. AskMSR generates an average of 7 rewrites per query (in the test collections to be described below). Largescale deployment of the system to many simultaneous users would place a significant burden on backend search engines employed to gather raw material for composing answers.


QA Enhanced Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address Go Links
AskMSR
Automatic Answer Finding Using the Web
Submit Query
Local intranet

Question: What does AAAI stand for?

AskMSR's results for the question: "*What does AAAI stand for?*":

American Association for Artificial Intelligence (12%)
Spring Symposium (8%)
Proceedings Template (7%)
Press (7%)
American (7%)
conference (6%)

Question: Who is Eric Brill?

AskMSR's results for the question: "*Who is Eric Brill?*":

a researcher (20%)
Natural Language Processing (19%)
tagger (14%)
speech (12%)
Machine Learning (12%)
science (11%)

Figure 2. User interface of AskMSR showing rank-ordered results for questions, "What does AAAI stand for?" and "Who is Eric Brill?"

We set out to explore the possibility of using machine learning to better understand the value of different kinds of rewrites, and to build models that could be used to control the classes and numbers of query rewrites issued to search engines by AskMSR. This work involves understanding how the probability of identifying a correct answer is influenced by the properties of the question and the nature and number of queries issued to search engines.

In addition to providing guidance on the policies for generating and submitting queries, models of accuracy and cost could enable the system to know when it would be best to skip completing the pursuit of an answer to a question. In these cases, the system could instead ask a user to attempt a reformulation of the question, or to seek the answer elsewhere. Beyond seeking characterization and control, probabilistic analyses of accuracy and value of alternate query policies could also lead to new insights with implications for refining the methods used by the base system.

Brill et al. (2002) explored a related problem of using learning techniques to estimate the confidence the system has in an answer. However, Brill et al. did not explore the quality of individual rewrites, the quantity of rewrites allowed, or perform a cost-benefit analysis as we have in this study.

5 Analysis of Answer Quality

Research on the analysis and control of the heuristic processes of the AskMSR system is facilitated by the system's architecture. AskMSR processes a question in distinct stages of a question-answering pipeline that can be analyzed independently. We set out to learn about the query reformulation and *n*-gram mining stages of the pipeline, with an eye on controlling the nature and numbers of queries issued to search engines.

5.1 Understanding the Value of Queries

In the pursuit of limiting the number of queries issued by AskMSR, we sought to replace the expert-derived heuristic functions used in AskMSR with Bayesian models that could generate probabilities of success for various rewrites. In an initial phase of analysis, we explored models that could provide a ranking of individual query rewrites. Our work on developing scores for query rewrites was stimulated by our direct inspection of the rewrites generated by the system, many of which appeared to be nonsensical or redundant. We sought to endow AskMSR with insights about poor query rewrites. We employed Bayesian learning procedures to generate models from a set of training cases that could be used to infer the probabilistic lift in accuracy that queries of different types would confer. Such models promised to provide a normalized metric for ordering sets of queries by their value, providing a decision surface for deliberating about the costs and benefits of using different numbers and types of query rewrites in a more global analysis of the end-to-end performance of the overall AskMSR system.

5.2 Establishing a Query-Quality Gradient

We first separated queries into two categories: (1) queries that involve ANDing of individual words and occasionally short phrases (e.g., population AND "of Japan"), and (2) queries that contain a single phrase (e.g., "the population of Japan is"). We refer to the former as *conjunctional rewrites*. We refer to the latter as *phrasal rewrites*. These two sets of queries have several distinct features, which we considered in our modeling efforts.

5.2.1 Formulation of Features

For both types of rewrites, we formulated features in an exploratory manner, defining evidential observations via intuition conditioned on knowledge about the distinctions that we could gain easy access to. Basic observations include the number of distinct words and the number and percentage of stop words present in the queries. For building predictive models of the goodness of *phrasal rewrites* we additionally examined features derived from a statistical natural language parser for English text created by the Natural Language Group at Microsoft. The syntactic parser constructs multiple parse trees, capturing multiple hypotheses for an input string, based on a consideration of the likely different parts of speech that words in a phrase can have. After producing all hypotheses, the parser employs a language model to rank the likely syntactic hypothesis, computing probabilities of each parse tree as the product of the probability of all of the nodes in the tree. The application of NLP parsing to each query rewrite does not put a significant computational burden on clients hosting AskMSR. Rewrites are parsed on an order of milliseconds.

We took into consideration several features output by the parser including the number of primary and secondary parses and the likelihood assigned to the maximum probability parse tree, a measure of the grammatical "goodness" of a query rewrite. A complete list of the features used for both sets of query rewrites is listed in Tables 1 and 2.

5.2.2 Learning Bayesian Networks from Data

To construct predictive Bayesian network models for inferring the best ordering of queries, we employed Bayesian model-structure learning. Such methods generate Bayesian networks, which highlight dependencies among variables and key influences on a target variable of interest. Over the last 10 years, methods for learning Bayesian networks from data have been refined (Cooper & Herskovits, 1992; Spiegelhalter, Dawid, Lauritzen & Cowell, 1993; Heckerman, Geiger & Chickering, 1995). Given a dataset, Bayesian-network learning methods perform heuristic search over a space of dependency models and employ a Bayesian model score to identify models with the greatest ability to predict the data. The Bayesian score estimates the likelihood of a model given data, $p(\text{model}|\text{data})$ by approximating the quantity, $p(\text{data}|\text{model}) * p(\text{model})$. Beyond the overall dependency structure and parameters of a Bayesian network, Chickering et

Table 1: Features of conjunctional and phrasal rewrites considered in learning models of query goodness.

LONGPHRASE: The longest phrase in the rewrite, in terms of words.
LONGWD: The length of the longest word in the entire query.
NUMCAP: The number of capitalized words in the entire query.
NUMPHRASES: The total number of phrases in the overall query.
NUMSTOP: The number of stop words in the entire query.
NUMWORDS: The number of words in the entire query string.
PCTSTOP: Percentage of stop words.
FILTER: The filter applied to the original query, such as “nlpwin_who_filter."
FILTER2 : Filters that focus on words and bigrams.
MAXRULE: Scores assigned at the query rewrite stage, based on the filter used to generate rewrites. This is the highest score procured for a particular query.

Table 2: Additional features used for learning models for phrasal rewrites.

PRIMARY_PARSES: The number of primary parses given by the natural language parser.
SECONDARY_PARSES: The number of secondary parses given by the natural language parser.
SGM: The “statistical goodness” of the rewrite; a measure of how grammatical the sentence or phrase is, given by the parser.

al. (1997) have developed methods for representing conditional probability distributions encoded within the variables (nodes) of Bayesian networks as decision graphs. These decision graphs, representing the conditional probability distributions of each node, are a generalization of decision trees since non-root nodes may have multiple parents. The dependency graphs can enhance the tractability of the representation by capturing asymmetries in the probabilistic influence of random variables upon one another. Representing such asymmetries can significantly reduce the size of the conditional probability tables represented within nodes of a Bayesian network, which would otherwise grow as the product of the state space of the parent variables. The methods of Chickering, Heckerman and Meek (1997) provide a learned Bayesian network as well as decision graphs underlying each variable in the network, elucidating the potentially asymmetric branching structure of predecessor variables and states.

5.2.3 Building Models for Predicting Success of Queries

We employed a system developed by Chickering et al. (1997) to build Bayesian networks from a training set consisting of cases of single queries and answers labeled by their correctness. To generate training cases, we ran AskMSR on 500 questions included in the TREC-9 data set. This data set includes a set of questions and correct answers used in the annual TREC workshop for evaluating the performance of competing question-answering systems (Voorhees & Harman, 2001). For each query, we collected rewrites generated by AskMSR. Cases were created by examining the features of conjunctional and phrasal query rewrites provided by the system (as shown in Tables 1 and 2), and noting the success of the system in answering the questions with single queries. The accuracy of the models was tested on a completely new set of 499 questions drawn from TREC-10 data set.

Figure 3 displays a learned Bayesian network for conjunctional rewrites. As represented by the arcs pointing into the variable labeled *The Score*, capturing the probability that the answer provided by AskMSR will be correct, the correctness of answers is influenced directly by the number of capital words, the longest phrase, the number of stop words, the longest word, and the percentage of stop words in the query rewrite. Other observables influence the probability that answers will be correct via dependencies with these variables. Figure 4 displays a decision tree associated with the Bayesian network model that shows the influence of properties of conjunctional rewrites on the expected accuracy of answers to questions. Figure 5 shows a Bayesian network that models the accuracy of answers associated with the issuance of queries derived from phrasal rewrites. In this model, beyond some of the variables that appeared as direct influencers of the accuracy of answers for the conjunctional rewrites, a variable representing the statistical goodness of the query rewrite is identified as having direct influence on the likelihood of retrieving a correct answer. The decision tree associated with this Bayesian network, representing, in a compact manner, the likelihood that an accurate answer will generated, is displayed in Figure 6. In summary, the two Bayesian models provide inferences about the probabilities that specific single rewrites will lead to a correct answer to a question. Given a set of query rewrites, we use the inferred probabilities that individual queries will achieve a correct answer as a *query-quality* score for ordering the list of rewrites in a subsequent analysis. The ordering provides a decision surface for a cost-benefit analysis of the ideal number of queries to issue. Although we have used probabilistic methods, we take the ordering as a heuristic approximation in that the system does not use single queries in normal operation, but rather ensembles of queries.

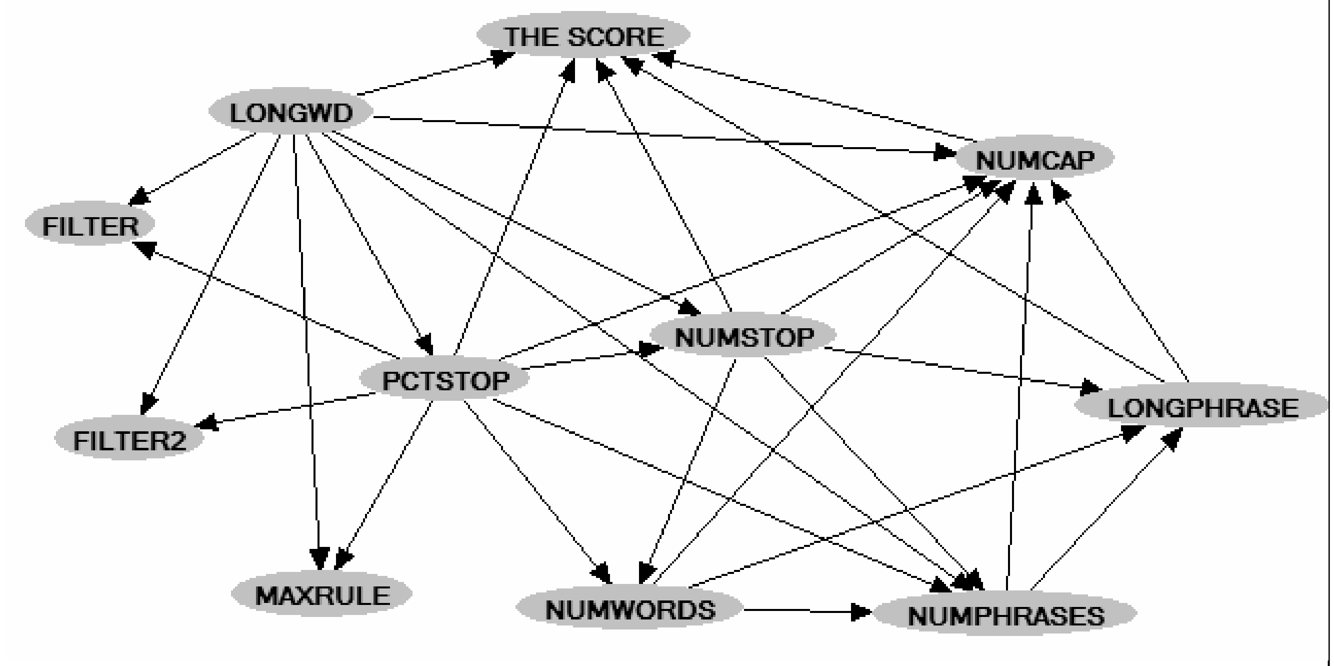


Figure 3. A learned Bayesian network showing dependencies among observations about the initial question and generated rewrites and the likelihood that a system will provide a correct answer for conjunctional rewrites.

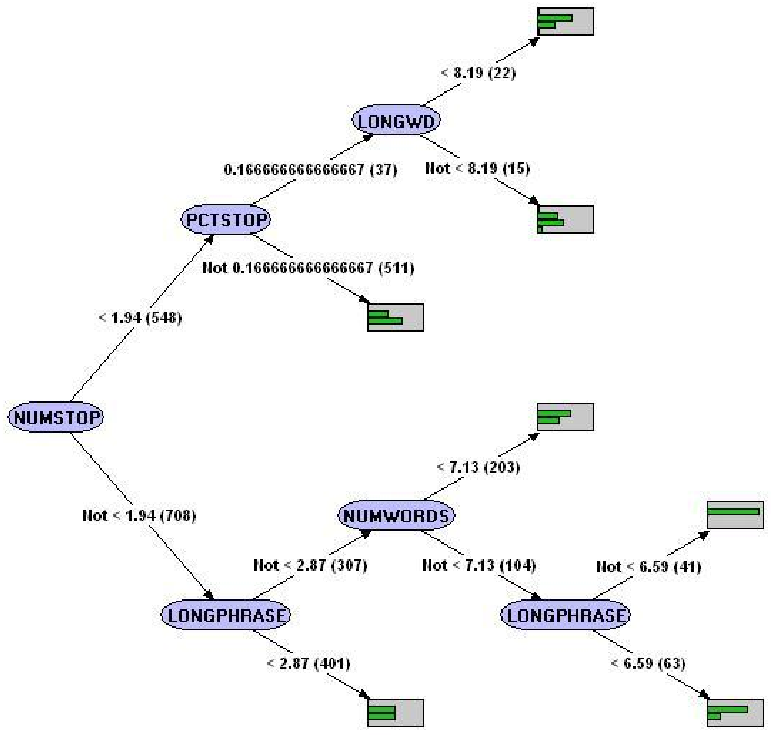


Figure 4: Decision tree encoding the conditional probability distribution within the *Score* node of the Bayesian network in Figure 3.

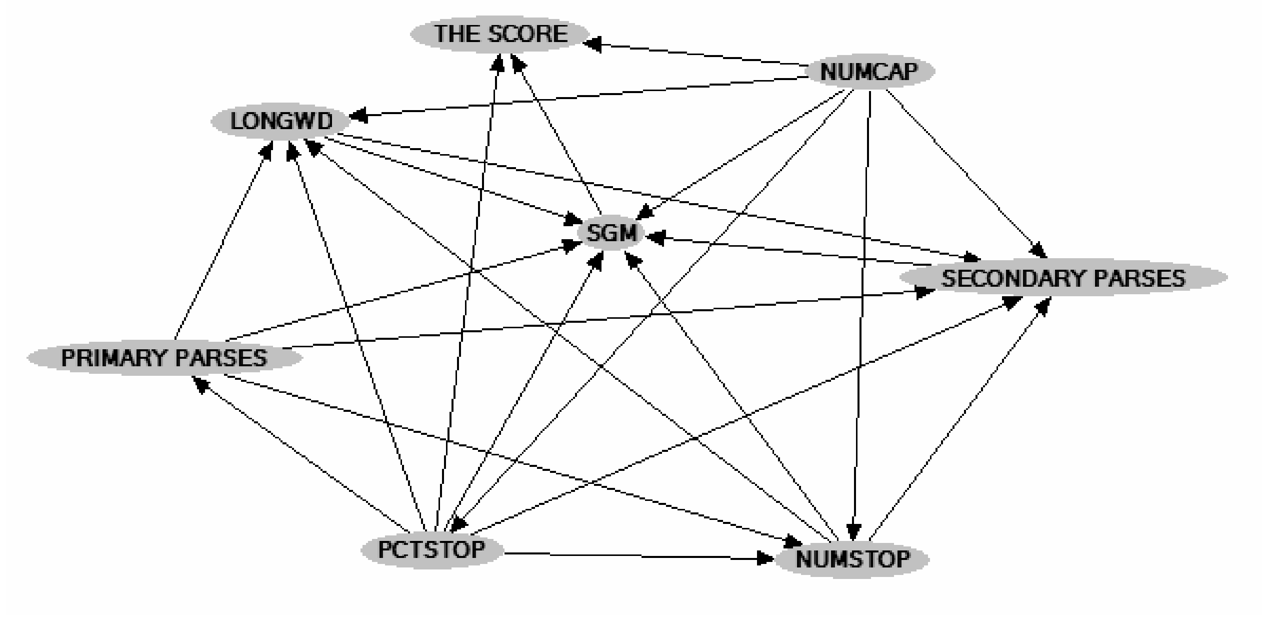


Figure 5. A learned Bayesian network showing dependencies among observations and the likelihood of correct answer for phrasal rewrites.

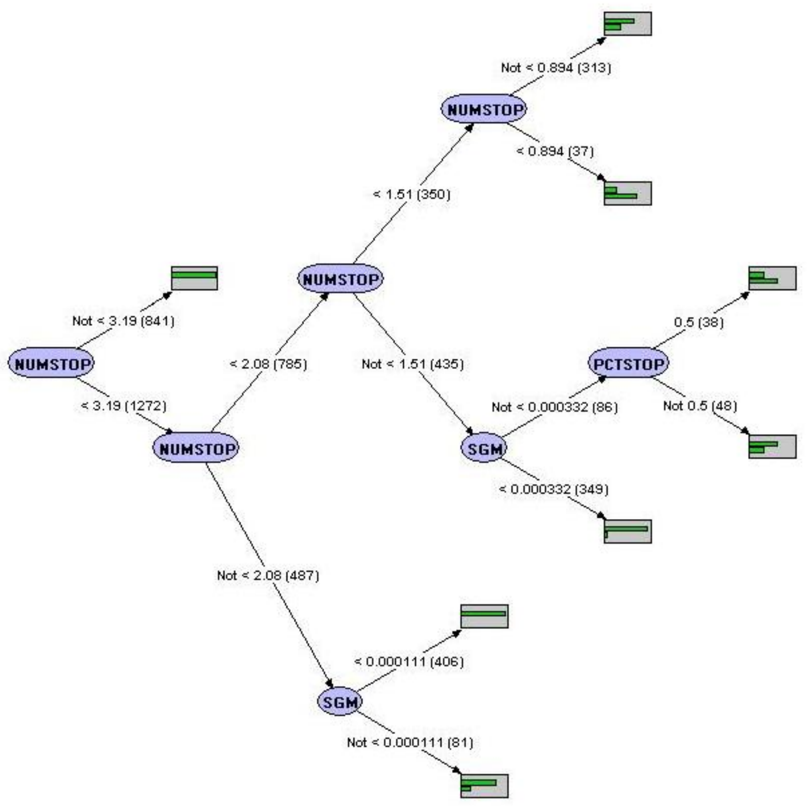


Figure 6: Decision tree for the conditional probability distributions for the *Score* variable of the Bayesian network displayed in Figure 5, representing the likelihood of generating a correct answer for phrasal queries, given observed values of ancestor variables in the Bayesian network.

5.3 Learning the Influence of Quantity on Quality

The initial analysis, yielding models of the usefulness of individual rewrites, enabled us to build a new version of AskMSR that orders the submission of queries according to the probability that individual queries will provide an accurate answer. In a second stage of learning and analysis, we set out to understand how best to control the numbers of queries issued by the revised version of AskMSR. In this phase of analyses, we again use machine learning to build Bayesian models of the relationship between the ultimate accuracy of AskMSR's processing of questions and the numbers of queries submitted to a search engine, given the properties of the question at hand. Such models enable cost-benefit analyses, trading off the expected gains in accuracy of an answer with the costs of submitting additional queries. These analyses provide AskMSR with new abilities for making dynamic decisions about the number of queries to submit to a search service—and to make decisions about when to forego an analysis and, instead, to ask a user to reformulate their question.

We built an ensemble of models by generating cases via a process of running AskMSR on TREC questions and applying different fixed thresholds on the number of rewrites submitted to search engines, as ordered by the goodness of queries established in the first phase of model construction. Additional features used in this phase are shown in Table 3. These features encode characteristics of the set of queries issued and the set of snippets returned. Snippets are the summaries

Table 3: Features considered by the models for inferring the accuracy of answers as a function of the numbers of rewrites issued.

AVERAGE_SNIPPETS_PER_REWRITE: The average number of snippets for all rewrites.
DIFF_SCORES_1_2: The difference between the first and second highest scored answer from AskMSR's scoring heuristic.
NUMNGRAMS: Total number of n-grams mined from snippets.
RULESCORE_X: Number of n-grams for rules with score X.
STD_DEVIATION_ANSWER_SCORES: The standard deviation among the top five answer scores from AskMSR's heuristic.
TOTALQUERIES: Total queries issued after all rewrites.
TOTNONBAGSNIPS: Total snippets generated from *phrasal* rewrites.
TOTSNIPS: Total snippets for all rewrites.

collected from Web pages for a given query

We note that the threshold numbers of rewrites were not always submitted because some questions generated fewer rewrites than the threshold values allowed. In our experiments, we discretized the number of queries into fixed thresholds at 1-10, 12, 15, and 20 rewrites per question, thus building 13 models. The models generated by this process provide predictions about the overall accuracy of answers to questions at increasingly higher thresholds on the numbers of query rewrites submitted to a back-end search engine.

Figure 7 displays a Bayesian network inferred for predicting the correctness of answers from observations about the initial question and the rewrites for the case where we limit submitted queries to 10 rewrites, as sorted by the initial ordering analysis. Figure 8 displays the corresponding decision tree learned from data about the performance of question answering in response to the 10-query limitation.

6 Cost-Benefit Considerations

Once we generate a set of Bayesian models that can predict the ultimate accuracy of answers to questions for different numbers of query rewrites, we are poised to deploy a system with the ability to dynamically control the number of queries used to answer previously unseen questions. We refer to the new version of AskMSR as AskMSR-DT (for AskMSR-*Decision Theoretic*).

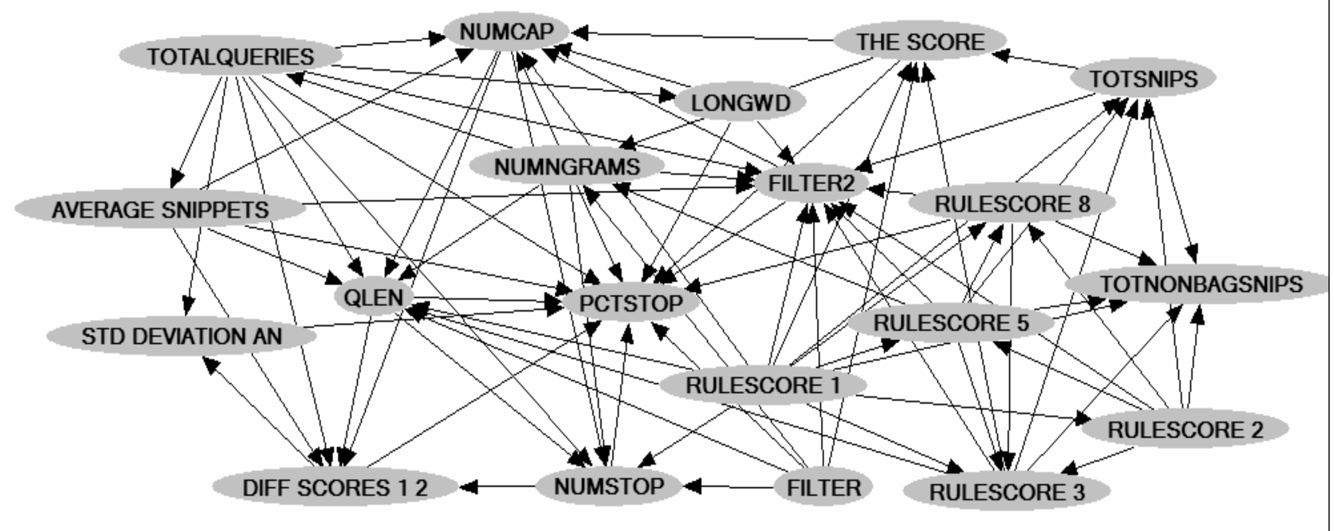


Figure 7. Bayesian network learned for query rewrite threshold of 10 rewrites per question. Bayesian network models such as this were constructed for 1-10, 12, 15, and 20 maximum rewrites per question.

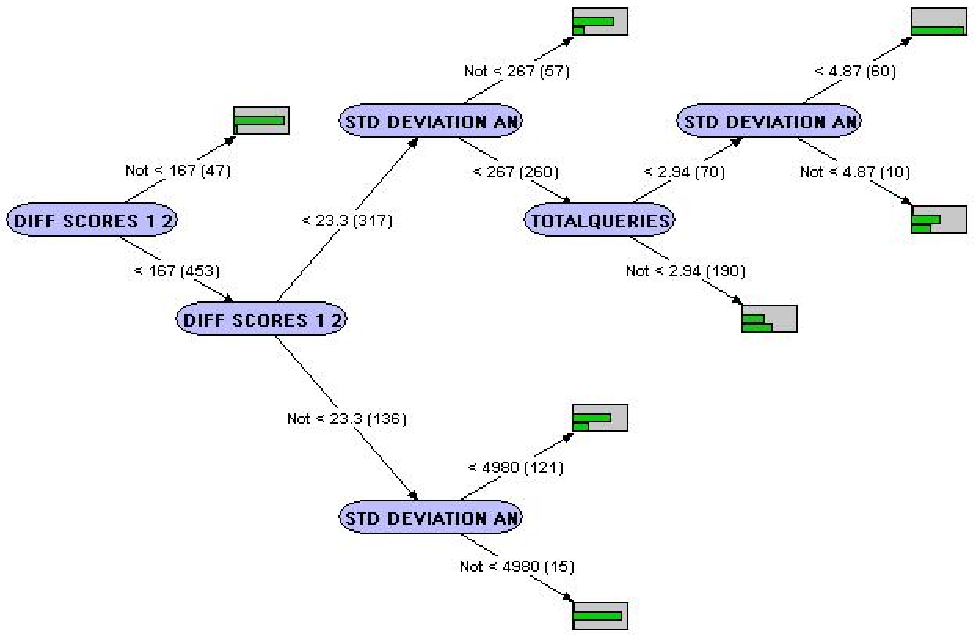


Figure 8: Decision tree for the Bayesian network predicting the accuracy of using a query rewrite threshold of 10 rewrites per question.

In controlling the number of queries relayed to a search engine, we need to represent preferences about the costs of sending increasing numbers of queries to search engines and the benefits of obtaining a more accurate answer. Several models for representing costs and benefits are feasible. In the general case, *costs* can be considered as a multiattribute vector of scarce commodities, including the dollar equivalent of the total investment in time required by a user, the computational cost associated with client activity to do processing, costs of a connection to the Internet required for communicating with a search engine, and the cost of queries to a search engine. We have focused on the cost represented by the number of queries issued for each question as this is the primary bottleneck in fielding a real-world version of a question-answering system like AskMSR. Question-answering systems based on re-writes pose a potentially significant challenge to search providers, as each query can lead to the generation of many individual queries, placing unbearable loads on a question-answering server that is scaled up in a real-world service, potentially addressing many millions of questions per day. In contrast, the computational costs of performing client-side inference, term counting, sorting, and cost-benefit analysis associated with AskMSR-DT—especially with the execution of preconstructed decision trees—is negligible. Nevertheless, we note that computational costs could become a significant consideration in variants of AskMSR or other systems performing more sophisticated real-time reasoning. For example, it is feasible that a solution might employ costly detailed parsing and statistical analysis of corpora. In such cases, we can incorporate computational costs by mapping both queries and computational costs to the same monetary scale in guiding a cost-benefit analysis.

We considered a model where a user or system designer assesses a parameter v, indicating the dollar value of receiving a correct answer to a question, and a parameter c representing the cost of each query rewrite submitted to a search engine. In an extension to the preference model, rather than asserting a constant value for receiving an answer to a question, the system could encode users' preferences about the value of receiving an answer as a function of the details of the situation at hand. For example, the value of an answer may be linked to the type of question, goals, location, and even the time of day for a user. Likewise the cost of submitting queries can be a function of such factors as the current load sensed on a search engine or the numbers of queries being submitted by a user's entire organization to a third-party search service. Costs may also be asserted directly as a fee for query by a search service. The costs may be linear in the number of queries or may scale non-linearly with increasing numbers of queries. For example, the first n queries may be considered free by a search service supporting the question-answering systems at an enterprise, after which expenses are incurred in a super-linear manner.

Models that output the probability of retrieving a successful answer, conditioned on different numbers of query rewrites, allow us to compute the expected value of submitting the queries. If we take the value of not receiving a valid answer as zero, the expected value of submitting n queries is the product of the likelihood of the answer, given evidence E about the query and background state of knowledge ξ, $p(A|E,n,\xi)$, and the value of obtaining a correct answer v, $p(A|E,n,\xi)\,v$.

In a deployed version of AskMSR-DT, a user or system administrator for an enterprise could be provided with an easy-to-use interface for assessing preferences about value and costs. Easy access to such controls would allow users to change preferences about the willingness to pay for accuracy in different settings.

Let us assume a utility model that can be decomposed into costs and benefits of outcomes. We shall explore the simple example of a preference model where the value of an answer, v, is assessed in terms of the cost of queries, c. That is, we assess the value of answers as some multiple k of the cost of each query c, $v=kc$. Assume a cost model that grows linearly with the number of queries, nc. In making decisions about the ideal number of queries to submit, we seek to optimize the net expected value, computed as the difference of the expected value and cost, for different n. Thus we wish to find the ideal number of queries, n^* where

$$n^* = \arg\max_n p(A|E,n,\xi)kc - nc \qquad (1)$$

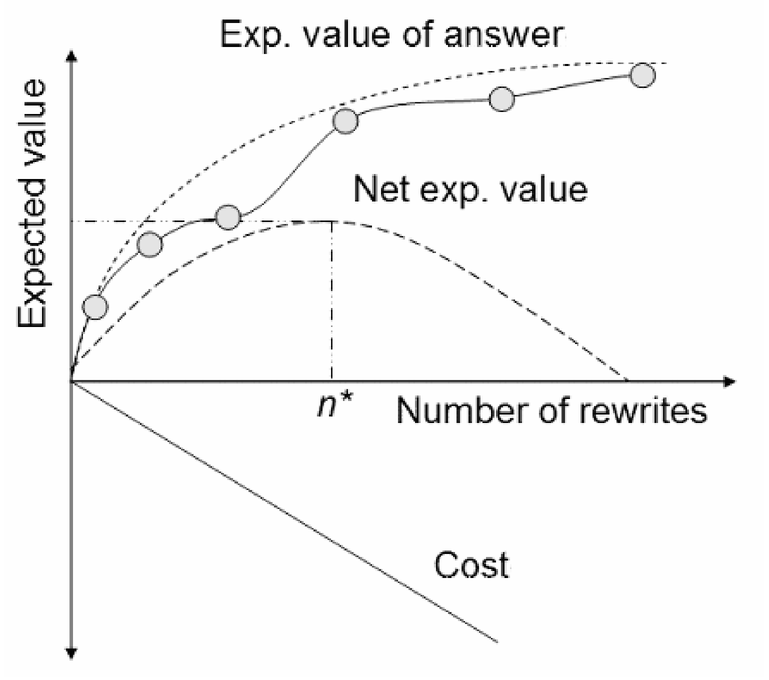


Figure 9: Key relationships in an idealized cost-benefit model for the case of decreasing returns in expected value of an answer with additional queries. The reality of non-monotonicity in expected value is highlighted with an irregular curve.

AskMSR-DT has the ability to check each quantity of query rewrites explored in the machine learning studies, and identify the best number of queries to submit.

Figure 9 shows an idealized view of the case of a cost-benefit analysis where the probability of an answer grows with decreasing marginal returns with additional query reformulations. In the idealized model, the expected value of the result increases with positive first derivative and a negative second derivative. The expected value, cost, and net expected value are displayed as a function of the number of queries submitted. If we had such smooth decreasing marginal returns on accuracy with increasing numbers of queries, we could identify *n** simply from the derivatives of the curves. As indicated in Figure 9, the ideal number of queries to issue is obtained at a position on the *x*-axis where the change in expected value of the answer is equal to the cost of each query.

In reality, the curve for the expected value of an answer with increasing numbers of rewrites may contain regions where the second derivative is positive (accelerating returns), and, more generally, shows non-monotonicity in the expected value of answers with queries. Thus, to identify the ideal number of queries to issue, we check the net expected value associated with increasing numbers of query rewrites for the range of numbers of rewrites under consideration. We can apply the set of models, described in Section 5.3, for predicting quality as a function of numbers of rewrites, and issue the numbers of query rewrites predicted as having a maximal net value. As we shall see in Section 7, we can further develop a marginal-value analysis, taking into consideration information that has been drawn from previously issued rewrites in deliberating about the value of additional rewrites. That is, rather than choosing the best number of rewrites *a priori*, based on Bayesian models for distinct numbers of queries, we consider observational features and predictions made available via the analysis of the query rewrites that have already been issued, in considering the value of submitting additional rewrites.

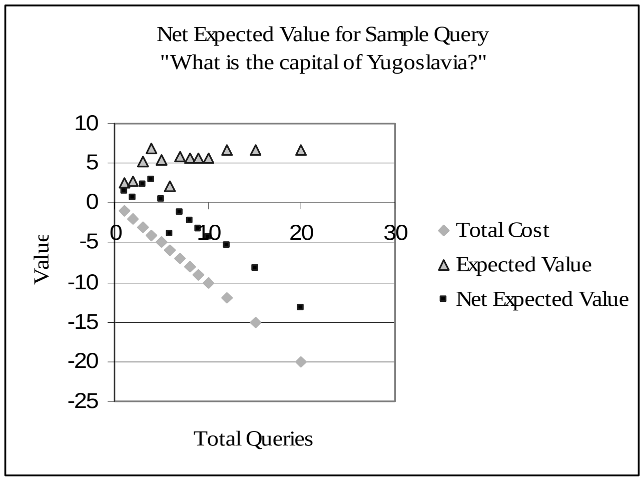
7 Empirical Studies of Decision Making

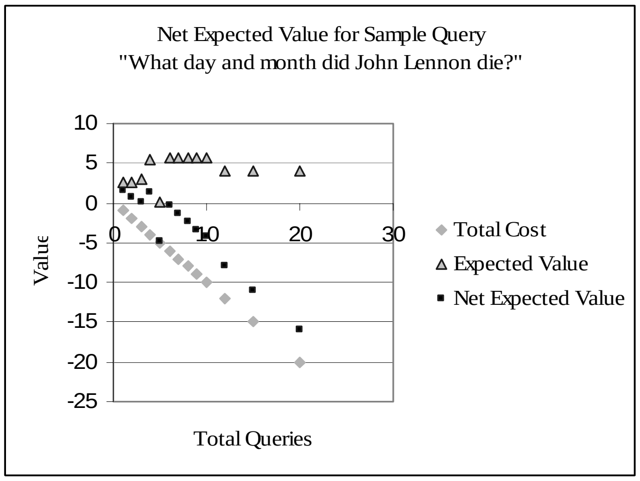
We performed a set of experiments with AskMSR-DT, employing the utility model described in Section 6, to drive dynamic decisions about the best number of query rewrites to select. Given a query, AskMSR-DT generates all rewrites that would have been submitted in the legacy AskMSR system. The query rewrites are first ranked by the single-query models. Then, we consider two different analyses in succession. In the first analysis, we perform inference with the ensemble of Bayesian models for different numbers of rewrites, employed in conjunction with the utility model, to select the ideal number of rewrites to issue to a search engine. In the second analysis, we explore the potential boosts in performance associated with a marginal-value method, that takes into consideration information made available from the analysis of previously submitted query rewrites, for a question-answering session.

7.1 *A Priori* Analysis of Ideal Numbers of Rewrites

Given the ensemble of Bayesian models constructed for different numbers of query rewrites, that we reviewed in Section 5, we can perform inference from features gleaned from a question to predict the probability of finding a successful answer with different numbers of rewrites. In a basic analysis, we consider a one-shot or *a priori* decision analysis of the ideal number of query rewrites to issue. That is, we assume that we must make a decision about the ideal number of rewrites to issue at the time a question is submitted to a system; we will not have the opportunity to make decisions in an incremental manner as interactions with a search engine progress. We use the predictions from the Bayesian models and the utility model to identify the most valuable number of queries to issue. The search results are then passed to the answer composition stage of the system. In the case of our studies, we consider different numbers of rewrites as represented in the predictive models which were trained for thresholds of 1-10, 12, 15, and 20 rewrites.

Figure 10a displays graphically key relationships of a cost-benefit analysis for the example query "*What is the capital of Yugoslavia?*," with a cost per query of 1 and a correct answer valued at 10. In this case, the best decision available is to





(a) (b)

Figure 10: (a) Expected value, cost, and net expected value of submitting different numbers of query rewrites for the question, "What is the capital of Yugoslavia?" (b) Expected value, cost, and net expected value of submitting different numbers of query rewrites for the question, "What day and month did John Lennon die?"

Table 4: Cost and accuracy for a set of baseline policies with constant cost per query.

Max Rewrites per question (*N*)	**Total Cost**	**Correct Answers, Random Order**	**Correct Answers, Query-Quality Order**	**Correct Answers, Conjunctional Rewrite and N-1 Random Order**	**Correct Answers, Conjunctional Rewrite and N-1 Query-Quality** Order
1	499	156	225	247	247
2	946	217	238	253	263
3	1383	243	254	264	267
4	1805	252	278	276	280
5	2186	272	282	276	282
6	2490	268	282	277	283
7	2738	272	282	281	283
8	2951	279	282	283	283
9	3103	276	282	282	283
10	3215	281	282	283	283
12	3334	281	283	282	283
15	3410	282	283	283	283
20	3426	283	283	283	283

choose to submit 4 query rewrites. Figure 10b displays the cost-benefit analysis for the query, "*What day and month did John Lennon die?*" for the same preference settings. The policy dictates that it is also best to send 4 query rewrites to the search engine.

Table 4 shows the performance of the system over different baseline policies. In these fixed-cost runs, AskMSR is given a ceiling on the number of query rewrites it can use. In the first set of experiments, the system chooses randomly from the rewrites available for each query up to a threshold (*N*). In a second set of experiments, AskMSR was executed with a static policy of selecting *N* rewrites from a list of query rewrites, ranked by the probabilistic query-quality score described in Section 5. Two additional baseline sets highlight the importance of the conjunctional rewrite, which is selected for each test run. For one set, the conjunctional rewrite is supplemented with randomly chosen rewrites up to the threshold (*N*). In the second, the policy is to select from the rewrite list ordered by probabilistic quality for the same set of thresholds. A ceiling of 20 rewrites is roughly equal to the policy in the legacy AskMSR system, which had no limitation on rewrites, as only a few queries yield more than 20 rewrites. As highlighted in the table of results, sequencing queries by the query-quality score dominates the randomly ordered queries, demonstrating the value of using the query-quality score. Figure 11 highlights the differences in accuracy for the four test sets over fixed thresholds of 1-5, 10. The

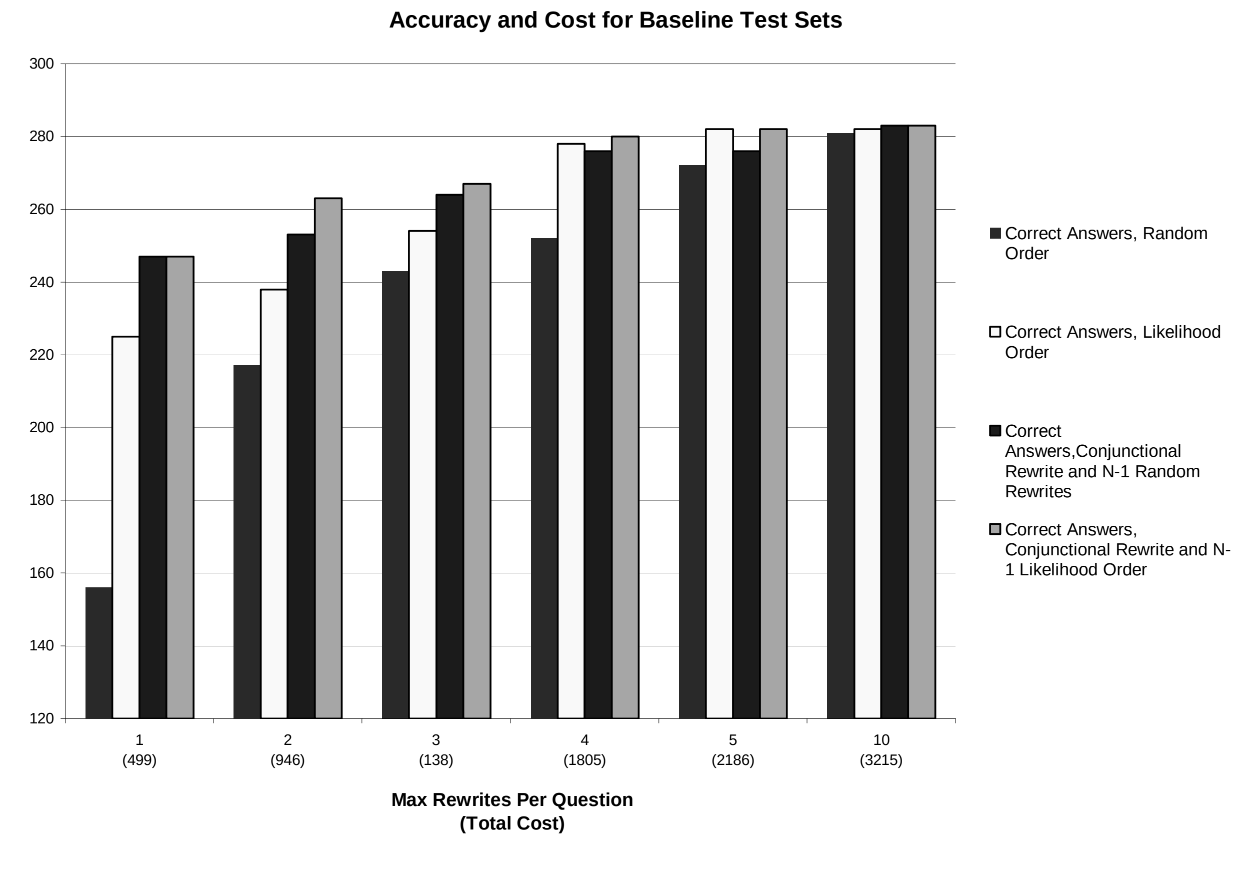


Figure 11: Cost and accuracy for over the four baseline analyses for thresholds of 1-5, 10. The maximum number of rewrites and the total cost, which is the same for each set, is listed on the *x* axis.

enormous influence on accuracy of the conjunctional rewrite, which returns a disproportionately large body of raw content due to its inherent generality, is displayed by the two sets that are hard-coded to select it every time.

The *a priori* cost-benefit analysis revealed that, for the current ASKMSR-DT system, selection of additional rewrites, beyond the conjunctional rewrite using likelihood ordering, is only slightly better than selecting sets of rewrites randomly, within each allowed number of queries. Nevertheless, the decision-theoretic methodology provides value by offering a rational, economic framework for making this evaluation, and an overall environment for the continuing refinement and evaluation of different rewrite selection policies. Our experiments with applying this framework, thus far, have identified that the likelihood ordering methodology does not overwhelm the conjunctional rewrites by a large margin. We found this to be an interesting result.

We also used the ranked query rewrites for cost-benefit analysis. Table 5 compares the policy chosen by the cost-benefit analysis with two fixed policies, one using only conjunctional rewrites (top row) and the other using all rewrites (bottom row). Our results show good performance for the system using the cost-benefit control (middle row). With the cost-benefit analysis, the system answers nearly as many correct as the original, unbounded system (277 versus 283), while posing less than a third of the total number of queries of those used in the absence of decision-theoretic guidance.

As a baseline comparison, the system was also executed with a fixed policy of using only the conjunctional rewrite for each question (first row, Table 5). This is useful because the conjunctional rewrite is the query reformulation that nearly always leads to the most results from the search-engine backend. This makes the conjunctional rewrite extremely valuable, as a greater set of intermediate results means a better chance of finding an answer. Our experiment shows that the conjunctional-query-only policy does fairly well, leading to 49% accuracy, using only 499 total queries (one query per question). However, this static policy is outperformed by the utility-directed system by a significant margin in terms of accuracy. Using the decision model, we achieve a 12% increase in correct answers at a cost of 680 additional queries. As another baseline, we considers the costs and performances of the current AskMSR system, which submits all rewrites. Table 6 shows the cost-benefit relationships for four different values of *k*. We note that, at a value of *k*=15, we reach the performance of the current AskMSR system but with many fewer queries (1346 vs. 3426).

Table 5: Cost and accuracy for AskMSR-DT using *a priori* decision analysis versus the static policies of choosing only the conjunctional rewrite and using all the rewrites.

Rewrite Policy	**Cost**	**Correct Answers (out of 499)**
Conjunctional rewrites only	499	247
Cost-benefit *k*=10, *c*=1	1179	277
All rewrites	3426	283

7.2 Experiments with Marginal Analyses

The analyses described in Section 7.1 consider the costs and benefits of submitting *N* query rewrites, independent of the outcome of previous rewrites for a question-answering session. We now consider an incremental, marginal analysis in which the outcome of the previous *n* rewrites informs the decision about whether to issue the next query rewrite *n+1*. Several variants of such marginal models are feasible. We shall focus on an approach where we build models that can predict the change in the probability of achieving a successful answer with the issuing of *n*+1 queries, considering several properties uncovered from analysis of the *n* rewrites already submitted.

As before, possible rewrites are ordered by the query-quality ordering described in Section 5, sequencing queries by the probability that individual rewrites will generate a correct answer. . Bayesian networks are then trained to predict the *Score* using as input the feature values for rewrite *n*, along with query features for rewrite *n+1*. With these trained models as guidance, we can make decisions about whether to add additional rewrites by identifying when the cost-benefit calculations show a drop in expected value with additional rewrites. Given potential non-monotonicities of the expected value curve, we explore the use of a lookahead parameter to bypass potential local maxima over a range of numbers of rewrites, referred to as the *lookahead horizon*. With a lookahead horizon of zero, ASKMSR-DT will stop adding rewrites when the predicted value of doing so first decreases. With a lookahead horizon of one or more, the system will continue to add rewrites as long as any rewrite within the horizon results in an increase in expected value. A lookahead of 12 represents full access to the expected value curve, as we considere 13 models (for *N*=1-10, 15, 20 query rewrites) in our experiments.

Table 7 shows the results of these experiments, with *k=10, c=1* and lookahead horizon of 0, 4, and 12. In addition, we show for comparison, models using only conjunctional rewrites and using all rewrites with full lookahead. We found that the marginal model provides benefits compared to using only the conjunctional rewrites. With a lookahead of 4, the system answers nearly as many questions correctly as the original, unbounded system (276 vs. 283), while posing fewer than a quarter of the number of queries (847 vs. 3426). We found that there are advantages to employing a lookahead of 4 compared to 0, because of the non-monotonicities in the expected-value curve. We discovered that further increases in lookahead horizon did not provide value. ASKMSR-DT, guided by the marginal analysis, answered nearly the same number of questions correctly (276 vs, 277) at much higher cost (847 vs. 1179) when the lookahead horizon was relaxed to include all rewrites.

Our investigations with marginal analyses highlight the value of building models that consider information made available via analysis of smaller number of rewrites in a session. To date, we have investigated only simple features of the previous *n* queries. We believe that there is promise in extending the predictive power of the marginal models by considering additional evidence, including richer, features based on multi-step summaries. Such features include observations that represent the trajectory of values of variables with increasing numbers of rewrites; there is an opportunity to collect data and to build predictive models that provide inferences about the likelihood that an answer will be correctly answered based on evidence returned for earlier subsets of rewrites submitted in real-time to search engines. New observations under consideration for such incremental analyses include the nature and distribution of text snippets or whole pages returned in response to earlier rewrite submissions, as well as evidence gleaned *about* the process of the ASKMSR-DT system attempting to piece together an answer from the results. We expect to see boosts in predictive power with the use of explicit marginal refinement models that are executed sequentially.

8 Summary and Future Work

We described a methodology for employing offline machine learning, and real-time probabilistic reasoning and decision analysis, for guiding search actions undertaken by a Web-based question-answering system. The methods show how we can bring rational economic guidance to the challenge of allocating search resources when seeking information from the large, ill-structured corpora reprepsented by the Web. We employed two phases of machine learning to build Bayesian models that predict the likelihood of generating an accurate answer to questions, and showed how we can couple such predictive models with considerations of the value and costs of issuing different numbers of queries to search engines. After reviewing a utility model that can inform a system about the costs and benefits of submitting additional query rewrites, we performed experiments with *a priori* and marginal analyses. Both methods showed how the expected value of answers can be traded for the cost of submitting increasing numbers of queries. We found that the marginal analyses provided boosts over the single-shot, a priori optimization for selecting the ideal number of queries.

As a research direction, we are interested in extending the decision-making considerations to consider prospects for harnessing mixed-initiative interaction (Horvitz, 1999), where the decision models consider real-time input from users to refine or reformulate questions. Beyond selecting the best web-querying actions to take, we can include in cost-benefit analyses a consideration of when it would be best to ask a user to reformulate a question rather than expending effort on handling a query that would be expensive or likely to yield inaccurate results. In such an analysis, we consider an assessment of the cost of delay and effort associated with a reformulation and the likelihood that a reformulation would lead to a better result.

We seek to boost the predictive power of the models of answer accuracy by considering additional features of questions and query rewrites. A promising area of research centers on extending the marginal analyses to consider richer information about the states of the question answering system, results returned previously from search engines during the session, and inferences from models for all earlier rewrites. We are also interested in the value of extending inference methods to acquire or reason about notions of topic, informational goals, and overall context of a user associated with a question. In relevant recent work, researchers learned models for predicting topic and high-level intentions associated with questions from tagged libraries of questions, posed by users of the Encarta online encyclopedia (Zukerman & Horvitz, 2001). The models provide predictions of the high-level information goals, topic, and desired level of detail of users, based on parts of speech and logical forms provided by an NLP parse of questions. There is opportunity to enhance the predictive power of models of the accuracy of answers to queries by incorporating distinctions and inferences explored in this prior work.

Beyond extending the probabilistic models of accuracy and expected value analysis, we are interested in refining the base question-answering system in several ways. Refinements of the base system could provide more power and control opportunities to the cost-benefit machinery. Promising refinements to the base system include introducing new variants of query rewrites and modifying methods for combining search results into candidate answers. A promising direction is the construction of predictive models for use in guiding the base-level answer-composition process. Such methods can serve as a means for guiding the counting and tiling heuristics currently used by AskMSR, or can provide new atomic composition methods. Methods for deliberating about different sources of information with probabilistic and decision-theoretic analyses may show promise in guiding the composition of answers from returned results, as they have for other tasks in information retrieval, such as in text classification based on multiple classifiers (Bennett, Dumais & Horvitz, 2002) and the prediction of retrieval quality from different sources (Nottelmann & Fuhr, 2003). In addition to guiding real-time question-answering procedures, the decision-analytic evaluative and control machinery can serve as a tool for developing insights and for guiding design,, enabling us to probe in an explicit manner the utility of making specific modifications to a base system.

We believe that pushing decision-theoretic analyses deeper into the operation of question-answering systems will be fruitful in the development of methods for taking advantage of the copious, but unstructured knowledge of the Web. Beyond question-answering systems, we suspect that similar methods for introducing a "normalizing layer" of probabilistic analysis of accuracy, coupled with utility-guided search and access, may be valuable for guiding the extraction of information from the Web and other large unstructured corpora for a spectrum of applications.

Acknowledgments

We thank David Weise for providing assistance with use of a statistical parser developed by the Natural Language Group at Microsoft.

References

Azari, D., Horvitz, E., Dumais, S. and Brill, E. (2003). Web-based question answering: A decision making perspective. In *Proceedings of the Conference on Uncertainty and Artificial Intelligence (UAI 2003),* pp. 11-19.

Brill, E., Dumais, S. and Banko, M. (2002). An analysis of the AskMSR question-answering system. In *Proceedings of 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002),* 257-264.

Bennett, P., Dumais, S., and Horvitz, E. (2002). Probabilistic combination of text classifiers using reliability indicators: Models and results. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (SIGIR 2002), pp. 207-214.

Chickering, D. M., Heckerman, D. and Meek, C. (1997). A Bayesian approach to learning Bayesian networks with local structure. In *Proceedings of Thirteenth Conference on Uncertainty in Artificial Intelligence,* pp. 80-89.

Clarke, C., Cormack, G. and Lyman, T. (2001). Exploiting redundancy in question answering. In *Proceedings of SIGIR'2001,* pp. 358-365.

Cooper, G. and Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning,* 9:309-347.

Dumais, S. T., Banko, M., Brill, E., Lin, J. and Ng, A. (2002). Web question answering: Is more always better? In *Proceedings of SIGIR'2002*, pp. 207-214.

Harabagiu, S., Moldovan, D., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., Girju, R., Rus, V. and Morarescu, P. (2001). FALCON: Boosting knowledge for question answering. In *Proceedings of the Ninth Text Retrieval Conference (TREC-9), NIST Special Publication 500-249*, pp. 479-488.

Heckerman, D., Geiger, D. and Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data, *Machine Learning,* 20:197-243.

Horvitz, E. (1999). Principles of mixed-initiative user interfaces. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pp. 159-166.

Kwok, C., Etzioni, O. and Weld, D. (2001). Scaling question answering to the Web. In *Proceedings of the 10th World Wide Web Conference (WWW'10),* pp. 150-161

Miller, G. (1995) WordNet: A Lexical Database. *Communication of the ACM*, 38(11):39-41.

Nottelmann, H. and Fuhr, N. (2003). From uncertain inference to probability of relevance for advanced IR applications. In *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR 2003),* pp. 235-250.

Spiegelhalter, D., Dawid, A., Lauritzen, S. and Cowell, R. (1993). Bayesian analysis in expert systems. *Statistical Science*, (8):219-282.

Voorhees, E. and Harman, D. (Eds.) (2001). *Proceedings of the Ninth Text Retrieval Conference (TREC-9), NIST Special Publication 500-249,* pp. 500-249.

Zukerman, I. and Horvitz, E. (2001). Using machine learning techniques to interpret WH-questions. In *Proceedings of Association for Computational Linguistics* (ACL-2001), pp. 547-554.