Finding the Right Facts in the Crowd: Factoid Question Answering over Social Media

Jiang Bian
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
jbian@cc.gatech.edu

Yandong Liu
Math & Computer Science Department
Emory University
Atlanta, GA 30332
yliu49@emory.edu

Eugene Agichtein
Math & Computer Science Department
Emory University
Atlanta, GA 30332
eugene@mathcs.emory.edu

Hongyuan Zha
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
zha@cc.gatech.edu

ABSTRACT

Community Question Answering has emerged as a popular and effective paradigm for a wide range of information needs. For example, to find out an obscure piece of trivia, it is now possible and even very effective to post a question on a popular community QA site such as Yahoo! Answers, and to rely on other users to provide answers, often within minutes. The importance of such community QA sites is magnified as they create archives of millions of questions and hundreds of millions of answers, many of which are invaluable for the information needs of other searchers. However, to make this immense body of knowledge accessible, effective answer retrieval is required. In particular, as any user can contribute an answer to a question, the majority of the content reflects personal, often unsubstantiated opinions. A ranking that combines both relevance and quality is required to make such archives usable for factual information retrieval. This task is challenging, as the structure and the contents of community QA archives differ significantly from the web setting. To address this problem we present a general ranking framework for factual information retrieval from social media. Results of a large scale evaluation demonstrate that our method is highly effective at retrieving well-formed, factual answers to questions, as evaluated on a standard factoid QA benchmark. We also show that our learning framework can be tuned with the minimum of manual labeling. Finally, we provide result analysis to gain deeper understanding of which features are significant for social media search and retrieval. Our system can be used as a crucial building block for combining results from a variety of social media content with general web search results, and to better integrate social media content for effective information access.

Categories and Subject Descriptors

H.3.3 [**Information Search Retrieval**]: Relevance feedback, Search process; H.3.5 [**On-line Information Services**]: Web-based services

General Terms

Algorithms Measurement Experimentation

Keywords

Community, Question Answering, Ranking

1. INTRODUCTION

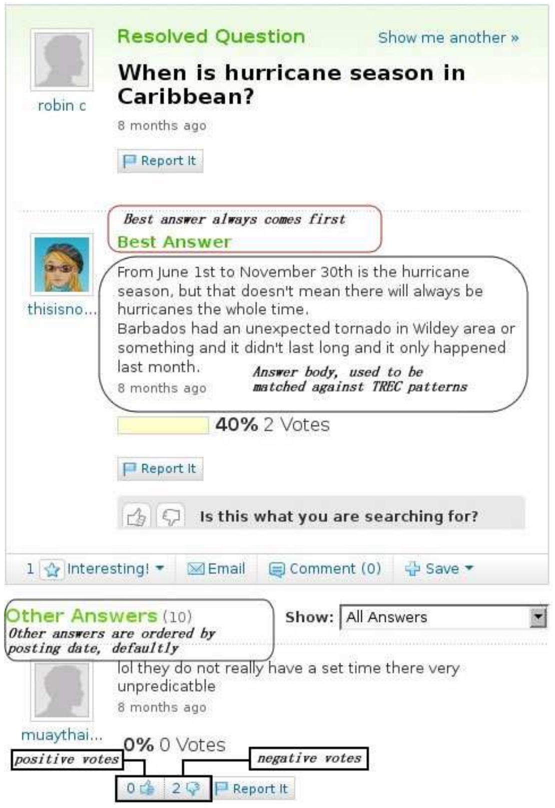
Online social media content and associated services comprise one of the fastest growing segments on the Web. Such content includes social bookmarking (Delicious[1]), social photo sharing (Flickr[2]), social video sharing (Youtube[3]), and many other forms of user-generated content. This content is different from the traditional content on the web (web pages) in style, quality, authorship, and explicit support for social graphs. The explicit support for social interactions between users, such as posting comments, rating content, and responding to questions and comments makes the social media unique and requires new techniques for analyzing and retrieving relevant content.

Question-Answering (henceforth QA) is a form of information retrieval where the users' information need is specified in the form of a natural language question, and the desired result is self-contained *answer* (not a list of documents). QA has been particularly amenable to social media, as it allows a potentially more effective alternative to web search by directly connecting users with the information needs to users willing to share the information directly. Some very successful examples of community sites organized around Question Answering are Yahoo! Answers[4], and Naver[5]. In these portals users can express specific information needs by posting questions, and get direct responses authored by other web users, rather than browsing results of search engines. Both questions and responses are stored for future use by allowing searchers to first attempt to locate an answer to their question, if the same or similar question has been answered in the

[1]http://del.icio.us/
[2]http://flickr.com/
[3]http://youtube.com/
[4]http://answers.yahoo.com/
[5]http://www.naver.com/

past. As QA portals grow in size and popularity, searching for existing answers for a given question becomes increasingly crucial to avoid duplication, and save time and effort for the users. For example, Yahoo! Answers now has tens of millions of users, and stores hundreds of millions of answers to previously asked questions. These databases of questions and respective answers is an invaluable resource for specific information needs not well served by general-purpose search engines.

In particular, today's search engines are not yet generally capable to automatically generate brief but precise summaries that integrate information from multiple sources, or to answer queries that require deep semantic understanding of the query or the document. For example, consider a query "When is the hurricane season in the Caribbean?", which we submit to both Yahoo! Answers and Yahoo! Web search engine. Figure 1 and 2 show the best answer and top 5 search results respectively. From those two figures, it is easy to see that Yahoo! Answers supply one brief answer but with high quality while for Yahoo! Search results people still need to click into the webpage to find needed information. Thus, QA sites and search services provides an alternative channel for obtaining information more quickly and more precisely on the web.


Resolved Question Show me another »

robin c

When is hurricane season in Caribbean?

8 months ago

Report It

Best answer always comes first

Best Answer

thisisno...

From June 1st to November 30th is the hurricane season, but that doesn't mean there will always be hurricanes the whole time.
Barbados had an unexpected tornado in Wildey area or something and it didn't last long and it only happened last month.

Answer body, used to be matched against TREC patterns

8 months ago

40% 2 Votes

Report It

Is this what you are searching for?

1 Interesting! Email Comment (0) Save

Other Answers (10)

Other answers are ordered by posting date, defaultly

Show: All Answers

lol they do not really have a set time there very unpredicatble

8 months ago

muaythai...

0% 0 Votes

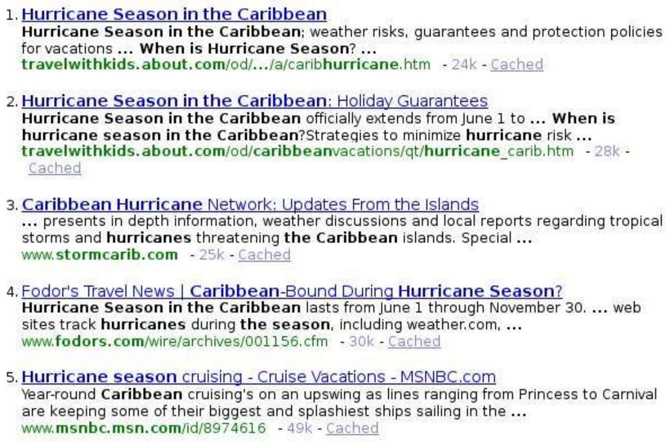
positive votes

negative votes

0 2 Report It

Figure 1: The question "When is hurricane season in Caribbean?" in Yahoo! Answer and its best answer.

However, finding relevant answers of a new query in QA archives is a difficult task that is distinct from web search and web question answering. The main differences are the availability of explicit user feedback and interactions, explicit authorship and attribution information, and organization of the content around topical categories and past question threads. Furthermore, the quality of the content varies even more than the quality of the traditional web content. A large fraction of the content reflects often unsubstantiated opinions of users, which are not useful for factual information retrieval and question answering. Hence, retrieving the correct factual answers to a question requires determining both relevance and quality of answer candidates.


1. Hurricane Season in the Caribbean
Hurricane Season in the Caribbean; weather risks, guarantees and protection policies for vacations ... When is Hurricane Season? ...
travelwithkids.about.com/od/.../a/caribhurricane.htm - 24k - Cached

2. Hurricane Season in the Caribbean: Holiday Guarantees
Hurricane Season in the Caribbean officially extends from June 1 to ... When is hurricane season in the Caribbean?Strategies to minimize hurricane risk ...
travelwithkids.about.com/od/caribbeanvacations/qt/hurricane_carib.htm - 28k - Cached

3. Caribbean Hurricane Network: Updates From the Islands
... presents in depth information, weather discussions and local reports regarding tropical storms and hurricanes threatening the Caribbean islands. Special ...
www.stormcarib.com - 25k - Cached

4. Fodor's Travel News | Caribbean-Bound During Hurricane Season?
Hurricane Season in the Caribbean lasts from June 1 through November 30. ... web sites track hurricanes during the season, including weather.com, ...
www.fodors.com/wire/archives/001156.cfm - 30k - Cached

5. Hurricane season cruising - Cruise Vacations - MSNBC.com
Year-round Caribbean cruising's on an upswing as lines ranging from Princess to Carnival are keeping some of their biggest and splashiest ships sailing in the ...
www.msnbc.msn.com/id/8974616 - 49k - Cached

Figure 2: Top 5 results when searching "When is hurricane season in Caribbean?" on Yahoo! Search.

As has been shown in recent work, user interactions, in particular explicit feedback from users, can provide a strong indication of the content quality [2]. Examples of such feedback include the selection of the "best" answer by the original question author, as well as the "thumbs up" and "thumbs down" ratings from other users. What is not clear from previous work is how to integrate explicit user feedback information and relevance into a unified ranking framework to retrieve answers that are relevant, factual, and of high quality.

In this paper we present a ranking framework to take advantage of user interaction information to retrieve high quality relevant content in social media. We focus on community Question Answering to explore how to integrate relevance, user interaction and community feedback information to find the right factual, well-formed content to answer a user's query. Our contributions in this paper include:

- An effective framework for learning ranking functions for question answering that incorporates community-based features and user preferences (Sections 3).
- Analysis of user interactions in community QA with insights into the relationship between answer quality, relevance, and factual accuracy (Section 5).

In summary, retrieving correct, factual, and well-formed answers from community QA archives is a crucial and challenging task. In this work we make significant progress towards this goal, allowing our system to find facts in the crowd with accuracy substantially higher than the current state-of-the-art.

2. RELATED WORK

Social media services provide popular web applications such as photo sharing (Flickr), social bookmarking (Delicious), and video sharing (Youtube), and, more recenlty, popular community Question Answering sites such as Yahoo! Answers and Naver. Question answering over community QA archives is different from traditional TREC QA [22], and applying QA techniques over the web [5]. The most significant difference is that traditional QA operates over a large collection of documents (and/or web pages) whereas we are attempting to retrieve answers from a social media archive with a large amount of associated user-generated metadata [2]. This metadata (such as explicit user feedback on answer quality) is crucial due to the large disparity of the answer quality, as any user is free to contribute his or her answer for any question.

Because of the explosive rise in popularity of Yahoo! Answers and Naver and other sites, community Question Answering has recently become an active area of research. This area of QA can be traced to the research on answering questions using Usenet Frequently Asked Questions (FAQ) archives [7, 19, 17, 20]. Usenet FAQs could be viewed as precursors to today's community QA archives that fulfilled a similar role but lacked intuitive support for explicit user feedback and other user interactions. More recently, Jeon et al. [10] presented retrieval methods based on machine translation models to find similar questions from Naver, a community QA service, but did not take quality of answers into consideration. Jurczyk and Agichtein [14] show an application of HITS algorithm to a QA portal. Zhang et al. [24] analyze data from an on-line forum, seeking to identify users with high expertise. Su et al. [21] analyzed the quality of answers in QA portals and found that the quality of each answers varies significantly. Jeon et al. [11] built a model for answer quality based on features derived from the specific answer being analyzed. Recently, Agichtein et al. [2] explored user interaction and content-based lexical features to identify high-quality content, but did not address how to retrieve *relevant* answers. In contrast, we propose a framework to integrate the interaction features into the answer retrieval.

Our work is similar in spirit to integrating user interactions and feedback into web search [12, 13, 15, 1]. For example, implicit feedback in the form of result clickthrough was shown to be helpful for web search ranking. The main difference of our current work is that we focus on question answering, which is a more precise form of search compared to general-purpose web search explored in the past. As another departure from previous work, we do not assume the existence of large amounts of expertly labeled relevance judgments, but instead automatically generate relevance labels. These differences in setting provide an interesting challenge for learning ranking functions.

Many models and methods have been proposed for designing ranking functions, including vector space models [3], probabilistic models [4] and recently developed language models [18]. Some advanced machine learning methods are also incorporated to learn ranking such as SVM and gradient boosting [12, 23]. In recent years, once relevance judgments are extracted from clickthrough data, several new learning frameworks, e.g. RankSVM [12], RankNet [6], RankBoost [8] have been created to utilize preference data to enhance ranking. Our work is also related to the recent research of Ko et al. [16] that propose a probabilistic framework for answer selection for traditional question answering.In this paper, we adapt a regression framework which is based on Gradient Boosting [25, 9]. We now present our ranking framework in more detail.

3. LEARNING RANKING FUNCTIONS FOR QA RETRIEVAL

Ranking functions are at the core of an effective QA retrieval system. In this section, we will explore a learning-based approach to the design of the ranking functions. We will focus on the specific characteristics of Yahoo! Answers and discuss how to employ user interactions and community-based features in Yahoo! Answers to rank the answers. We start with a more precise definition of the problem of QA retrieval, and then describe the different interactions available in a state-of-the-art community QA portal. Then we discuss how to represent textual elements and community-based elements in Yahoo! Answers as a vector of features. We then show how to extract preference data on answers from user interactions with Yahoo! Answers. Based on the extracted features and preference data, we apply the regression-based gradient boosting framework [25] to the problem of learning ranking function for QA retrieval.

3.1 Problem Definition of QA Retrieval

In QA systems, there are a very large amount of questions and answers posted by a diverse community of users. One posted question can solicitate several answers from a number of different users with varying degree of relevance to the posted question. We abstract the social content in QA system as a set of question-answer pairs:

$$\langle qst_i, ans_i^j \rangle$$

where qst_i is the ith question in the whole archive of the QA system and ans_i^j is the jth answer to this question.

Given a user query, our goal is to order the set of QA pairs according to their relevance to the query, and the ordering is done by learning a ranking function for triples of the form,

$$\langle qr_k, qst_i, ans_i^j \rangle,$$

where qr_k is the k-query in a set of queries. We will first discuss how to effectively extract features and preference data for each triple of the above form and then discuss a regression-based gradient boosting methods for learning a ranking function for QA retrieval.

3.2 User Interactions in Community QA

Community QA is a particularly popular form of social media, drawing millions of users to ask and answer each others' questions. Unlike other social media services, community QA services such as Yahoo! Answers provide distinct types of interactions among users that are specific to the QA domain. Users in Yahoo! Answers do not only ask and answer questions, but also actively participate in regulating the system. A user can vote for answers of other users, mark interesting questions and even report abusive behavior. Therefore, a Yahoo! Answers user has a threefold role: asker, answerer and evaluator. And there are respectively three types of user interaction activities: asking, answering and evaluating. We summarize elements and interactions in Yahoo! Answers in Figure 3. The rest of this paper will focus on how to utilize this information for QA retrieval.

In addition to facilitating the community question answering process, a community QA system must support effective search of the archives of past questions and answers. In fact, one benefit of Yahoo! Answers to users and web searchers is precisely the immense archive of hundreds of millions of answers to past questions, with the associated community feedback. Searching this archive allows users

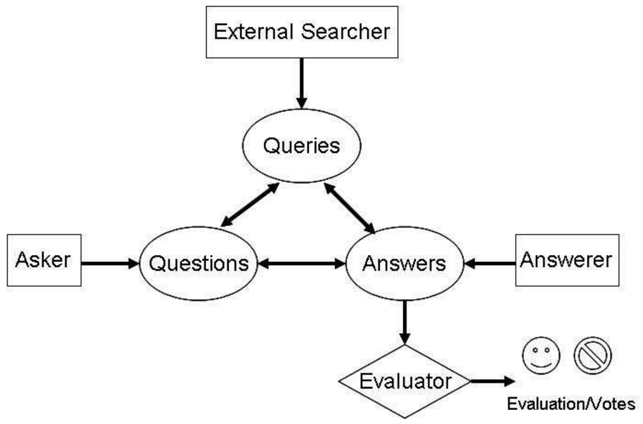


Figure 3: Elements and interactions in Yahoo! Answers: Askers post questions on Yahoo! Answers; Several users-answerers read questions and supply their answers to them; Users can also read these answers and give evaluations/votes; External users can submit queries to Yahoo! Answers and receive relevant questions with answers.

the benefit of community feedback and precise and specific answers for many information needs not supported by general web search. However, because of the disparity in answer quality, and the difficulty in mapping user information needs to past questions, the problem of retrieving both *relevant* and *high quality* factual answers requires the integration of both content features (to estimate relevance) as well as user interaction and community features (to estimate quality). Having introduced our setting, we now describe our features for answer retrieval.

3.3 Features and Preference Data Extraction

We follow the general practice in information retrieval and represent each query-question-answer triple (qr, qst, ans) as a combination of textual features (i.e., textual similarity between query, question and answers), statistical features (i.e., independent features for query, question and answers) and social features (i.e., user interaction activities and community-based elements). In Yahoo! Answers, there is an additional important type of user feedback — user evaluation in the form of votes (represented as the "thumbs up" and "thumbs down" metaphors). We can use this information to infer preference relevance judgments for the set of answers. In the following, we discuss features and preference data extraction in more details.

3.3.1 Representing Textual Elements and User Interactions as Features

Textual Elements Features: As we mentioned before, there are three textual elements in Yahoo! Answers: questions, answers and queries (showed in Figure 3). We first design features from each of these three elements independently, such as "number of tokens for a query" for query, "how long has the question been posted" for question, "number of received votes for an answer" for answer etc. (We describe them as statistical features in Table 1)

Then, we also extract textual features from relationship between questions, answers and queries. For example, the number of overlapping terms and token number ratio between two of these three elements. We also introduce other features describing similarity between these three elements. (We describe them as textual features in Table 1)

User Interaction Features: As discussed before, there are three kinds of roles each user in QA system may play, namely Asker, Answerer and Evaluator. Figure 3 shows the interactions between these three roles. Askers post their questions on QA system in the hope that other users answer these questions. Answerers submit their answers to the question in the QA system. Some answers have high quality while the majority are not useful. Evaluators give positive and negative votes for an answer after they read an answer and related question in the QA system. In the community of QA system, each user can play all of these three roles at the same time.

For each user in the user community of a QA system, there are several features to describe his or her activities, such as the number of questions he or she asked, the number of answers he or she posted, the number of best answers he or she posted etc. These features to certain extent can approximate the user's expertise in the QA community. And user's expertise within certain topics can in turn indicate the quality of his or her answers to the questions about certain topics. For example, in a query-question-answer triple, if answerer tend to post useful answers or even best answers in the past, he or she is more likely to give answers of high quality this time. Similarly reputation of askers and evaluators can also indicate quality of answers. Therefore, in each query-question-answer triple, we also extract features indicating user's activities in the QA system. As there is no information about evaluators in Yahoo! Answer, we only consider features for askers and answerers, such as "number of questions the asker asked in the community", "number of best answers the answerer posted in community". These features are listed in Table 1.

3.3.2 Representing Users Evaluations as Preference Data

One of the user interactions in a QA community, especially in Yahoo! Answers, is their evaluations for the existing answers. In Yahoo! Answers, after reading existing answers for a question, user can give his or her judgment as the evaluation for the answers. If he or she considers the answer as useful and of high quality, he or she can add a plus vote to this answer. Otherwise, a minus votes may be added to the answer.

We examine users evaluation data and extracted a set of preference data which can be used for ranking the answers as follows. For each query qr, under the same question qst, we consider two existing answers ans_1 and ans_2 from Yahoo! Answers. Assume that in the users evaluation data, ans_1 has p_1 plus votes and m_1 minus votes out of n_1 impressions while ans_2 has p_2 plus votes and m_2 minus votes out of n_2 impression. We want to consider answer pairs ans_1 and ans_2 to see whether ans_1 is preferred over ans_2 in terms of their relevance to the question qst. To this end, we assume that plus votes obey binomial distribution, showed as following:

$$B(k; n, p) = \begin{pmatrix} n \\ k \end{pmatrix} p^k (1-p)^{n-k}$$

We use the approach in [25] and apply likelihood ratio test to examine whether a pair of answers is significant or not, i.e., whether there are enough votes to compare the pair. In

Table 1: Features used to represent textual elements and user interactions

Textual Features of Questions, Answers and Queries	
Query-Question Overlap	Overlapping terms between query and question
Query-Answer Overlap	Overlapping terms between query and answer
Question-Answer Overlap	Overlapping terms between question and answer
Query-Question length ratio	Ratio between number of tokens in query and question
Query-Answer length ratio	Ratio between number of tokens in query and answer
Question-Answer length ratio	Ratio between number of tokens in question and answer
Statistical Features of Questions, Answers and Queries	
Query Length	Number of tokens in query
Question Length	Number of tokens in question
Answer Length	Number of tokens in answer
Question Lifetime	How long has this question been posted
Answer Lifetime	How long has this answer been posted
Yahoo! Question Rank	The rank of question in Yahoo! Answers
Yahoo! Answer Rank	The rank of answer in Yahoo! Answers
Question Popularity	How many answers are received under the question
Votes Number	Number of votes for answer
User Interaction/Social Elements Features	
Asker Total Points	Points calculated by Yahoo! based on Asker's activity history
Asker Total Answers	How many answers does the asker submit in Yahoo! Answers
Asker Best Answer	How many best answers does the asker propose in Yahoo! Answers
Number of Questions Asked by Asker	How many questions does the asker post in Yahoo! Answers
Number of Questions Resolved by Asker	How many questions are resolved by the asker in Yahoo! Answers
Asker stars received	How many stars does the asker receive in Yahoo! Answers
Answerer Total Points	Points calculated by Yahoo! based on Answerer's activity history
Answerer Total Answers	How many answers does the Answerer submit in Yahoo! Answers
Answerer Best Answer	How many best answers does the Answerer propose in Yahoo! Answers
Number of Questions Asked by Answerer	How many questions does the Answerer post in Yahoo! Answers
Number of Questions Resolved by Answerer	How many questions are resolved by the Answerer in Yahoo! Answers
Answerer stars received	How many stars does the Answerer receive in Yahoo! Answers

particular we compute the following statistic,

$$\lambda = \frac{B(p_1+p_2; n_1+n_2, (p_1+p_2)/(n_1+n_2))}{B(p_1; n_1, p_1/n_1)B(p_2; n_2, p_2/n_2)} \rightarrow -\chi^2$$

For a pair of answers ans_1 and ans_2, when the above value is greater than a threshold, we say the pair is significant. If ans_1 and ans_2 form a significant pair, we then extract a preference for the pair by comparing $\frac{p_1}{p_1+m_1+s}$ with $\frac{p_2}{p_2+m_2+s}$, where s is positive constant, i.e., if the former value is bigger than the later one, then we say ans_1 is preferred over ans_2 which is denoted by $ans_1 \succ ans_2$, and vice versa.

3.3.3 Representing Labeled data as Preference Data

Besides users evaluation information, we can also extract preference data from labeled data. For two query-question-answer items with the same query,

$$(qr, qst_1, ans_1) \quad (qr, qst_2, ans_2),$$

let their feature vectors be x and y, respectively. If ans_1 has a higher labeled grade than ans_2, we include the preference $x \succ y$ while if ans_2 has a higher labeled grade than ans_1, we include the preference $y \succ x$. We will discuss how to obtain labeled data in details in section 4.1.

3.4 Learning Ranking Function from Preference Data

Once the features and preference data are extracted, the next question is how to use them for the purpose of learning a ranking function for QA retrieval. We apply a framework for solving ranking problems from preference data developed in [25]. This framework proposes a squared hinge loss function for learning ranking functions from preference data; it also presents an algorithm that adapts functional gradient descent for minimizing the loss function. We now briefly describe the basic idea of the algorithm in [25].

Suppose the set of available preferences is

$$\mathcal{S} = \{\langle x_i, y_i \rangle \mid x_i \succ y_i, i = 1, ..., N\}.$$

Here each $\langle x, y \rangle \in S$, x, y denote the feature vectors for two query-question-answer triples with the same query. $x \succ y$ means that x is preferred over y, i.e. x should be ranked higher than y. In other words, the answer in x is considered more relevant than that in y with respect to the same query in both triples.

In [25], the problem of learning ranking functions is cast as the problem of computing a function h, such that h match the given set of preferences, i.e., $h(x_i) \geq h(y_i)$, if $x_i \succ y_i$, $i = 1, ..., N$, as much as possible. The following objective function (squared hinge loss function) is used to measure the risk of a given ranking function h,[6]

[6]This loss function can be considered as a smooth surrogate of the total number of contradicting pairs in the given preference data with respect to the function h. We say $x \succ y$ is a contradicting pair with respect to h if $h(x) < h(y)$.

$$\mathcal{R}(h) = \frac{1}{2}\sum_{i=1}^{N}(\max\{0, h(y_i) - h(x_i) + \tau\})^2,$$

and we need to solve the following minimization problem

$$\min_{h\in\mathcal{H}} \mathcal{R}(h),$$

where $\mathcal{H}$ is a function class, chosen to be linear combinations of regression trees in our case. The above minimization problem is solved by using functional gradient descent discussed in [9]. We summarize the algorithm for learning ranking function h using gradient boosting (GBrank) as follows:

Algorithm GBrank:
Start with an initial guess h_0, for $k = 1, 2, ...$

1. Using h_{k-1} as the current approximation of h, we separate $\mathcal{S}$ into two disjoint sets,
   $$\mathcal{S}^+ = \{\langle x_i, y_i\rangle \in \mathcal{S} | h_{k-1}(x_i) \geq h_{k-1}(y_i) + \tau\}$$
   and
   $$\mathcal{S}^- = \{\langle x_i, y_i\rangle \in \mathcal{S} | h_{k-1}(x_i) < h_{k-1}(y_i) + \tau\}$$
2. Fit a regression function $g_k(x)$ using Gradient Boosting Tree [9] and the following training data
   $$\{(x_i, h_{k-1}(y_i) + \tau), (y_i, h_{k-1}(x_i) - \tau) | \langle x_i, y_i\rangle \in \mathcal{S}^-\}$$
3. Form the new ranking function as
   $$h_k(x) = \frac{k h_{k-1}(x) + \eta g_k(x)}{k+1}$$
   where η is a shrinkage factor.

Two parameters need to be determined: the shrinkage factor and the number of iterations, this is usually done by cross-validation [25].

4. EXPERIMENTAL SETUP

This section presents our evaluation setup. First we describe our dataset including the queries and the corresponding corpus of questions and answers. Then we describe our evaluation metrics (Section 4.2) and the ranking methods to compare (Section 4.3) for the experimental results reported in Section 5.

4.1 Datasets

Factoid questions from the TREC QA benchmarks
We use factoid questions from seven years of the TREC QA track evaluations (years 1999–2006)[7] for the experiments reported in Section 6. It is worth noting that TREC questions from the years 1999 to 2003 are independent of each other: each question is self-contained and we submit directly as the query. Starting from 2004, however, the questions are organized in groups with a 'target'. For those questions, we submit their 'target' as well as the questions themselves. In total, approximately 3,000 factoid TREC questions were compiled as the initial set of queries.

Since we need *some* candidate answers from Yahoo! Answers to estimate how well different ranking functions perform, we select the 1250 TREC factoid questions that have at least one similar question in the Yahoo! Answers archive.

[7]http://trec.nist.gov/data/qa.html

Question-answer collection dataset
Our dataset was collected in order to simulate a user's experience with a community QA site. We submit each TREC query to the Yahoo! Answers web service[8] and retrieve up to 10 top-ranked related questions according to the Yahoo! Answers ranking. For each of these Yahoo! questions, we retrieve as many answers as there are available for each question thread. There are, in total, 89642 $\langle query, question, answer\rangle$ tuples. 17711 tuples (19.8%) are labeled as "relevant" while 71931 (81.2%) are labeled as non-relevant.

Relevance Judgments
In our experiment, the data are labeled in two ways: by using the TREC factoid answer patterns, and, independently, manually in order to validate the pattern-based automatic labels.

For automatic relevance labels we use the available regular expression answer patterns for the TREC factoid questions. We check every answer's text body, and if the text matches one of the answer patterns, we consider the answer text to be relevant, and non-relevant otherwise.

In order to validate the accuracy of our automatically-assigned relevance labels, we independently labeled a number of answers by hand. The manually labeled answers were compared with the automatically generated labels, resulting in over 90% agreement between the automatic and manual methods. In the cases of disagreements were due to the excessive strictness of the answer patterns, and to the world changing (e.g., with a different correct answer for a question "Who is the prime minister of Japan."). This is not surprising, as some of the answer patterns were created years ago around the time of the TREC QA evaluation. In summary, automatically generated labels, even though with some small degree of noise, nevertheless exhibit high agreement with manual relevance judgments, and serve as a good proxy for comparing rankings.

4.2 Evaluation Metrics

We adapt the following information retrieval metrics to evaluate the performance of the ranking function.

- **Mean Reciprocal Rank(MRR)**: The MRR of each individual query is the reciprocal of the rank at which the first relevant answer was returned, or 0 if none of the top N results contained a relevant answer.The score for a sequence of queries is the mean of the individual query's reciprocal ranks. Thus, MRR is calculated as
  $$MRR = \frac{1}{|Qr|}\sum_{q\in Qr}\frac{1}{r_q}$$
  where Qr is a set of test queries, r_q is the rank of the first relevant document for q.
- **Precision at K**: for a given query, $P(K)$ reports the fraction of answers ranked in the top K results that are labeled as relevant. In our setting, we require a relevant answer to be labeled "matched" for TREC pattern. For this metric, the position of relevant answers within the top K is irrelevant, while it measures overall user potential satisfaction with the top K results.
- **Mean Average of Precision(MAP)**: Average precision for each query is defined as the mean of the precision at K values calculated after each relevant answers

[8]http://developer.yahoo.com/answers/

was retrieved. The final MAP value is defined as the mean of average precisions of all queries in the test set. This metrics is the most commonly used single-value summary of a run over a set of queries. Thus, MAP is calculated as

$$MAP = \frac{1}{|Qr|} \sum_{q \in Qr} \frac{\sum_{r=1}^{N}(P(r) \times rel(r))}{|R_q|}$$

where Qr is a set of test queries, R_q is the set of relevant document for q, r is the rank, N is the number retrieved, $rel()$ is a binary function on the relevance of a given rank, and $P()$ is precision at a given cut-off rank.

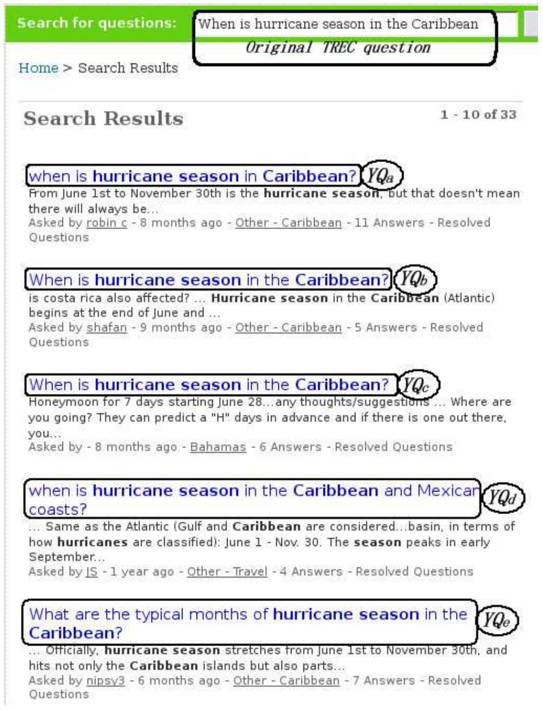
4.3 Ranking Methods Compared

To evaluate the Q&A retrieval quality, we compare the quality of following methods:

- Baseline_Yahoo(baseline1): In this method, we adapt default ranking in Yahoo! Answers. Answers to a particular question are ordered by posting date. The older one is ranked higher except that best answers always come first.
- Baseline_Votes(baseline2): In this method, similar to our baseline1, we let best answers always be on top of the answer list. However, following answers are ranked in decreasing order by number of (positive votes - negative votes) received. If there is no vote for some answers, we order them by Yahoo! default ranking.
- GBRanking: Ranking function with community/social features: this is our method presented in Section 3.4

For Yahoo! Answers, since we first get a list of Yahoo! questions for one TREC query, and each of these Yahoo! questions has its own answers, there are multiple alternatives for calculating MRR, Precision and MAP values for *Baseline_Yahoo* and *Baseline_Votes*. First, we need to introduce some nomenclature: for each TREC query TQ, we retrieve a list of related questions from Yahoo! Answers YQ_a, YQ_b... (as shown in Figure 4). After clicking on one of these questions, we get the answers to each question, e.g., YQ_a, as $YQ_a^1, YQ_a^2 ... YQ_a^n$, as shown in Figure 1:

- **MRR_MAX**: Calculate MRR value for each YQ_a, YQ_b... and use the highest value as this TQ's MRR. This baseline simulates an "intelligent" user who always selects the most relevant retrieved Yahoo! question thread first (as measured by the corresponding MRR for the thread).
- **MRR_STRICT**: Calculate MRR value for each YQ_a, YQ_b... and use the their average value as this TQ's MRR. This baseline simulates a user who blindly follows the Yahoo! Answer's ranking and examines retrieved question threads and corresponding answers in the order they were originally ranked.
- **MRR_RR**(round robin): Use YQ_a's first answer as TQ's first answer, YQ_b' first answer as TQ's second answer and so on, then calculate this TQ's MRR. This baseline simulates a "jumpy" user who believes that answers that come first, no matter to which questions, are always better, and jumps between question threads looking at the top-ranked answers for each tread in order of the original ranking.


Search for questions: When is hurricane season in the Caribbean

Original TREC question

Home > Search Results

Search Results 1 - 10 of 33

when is **hurricane season** in **Caribbean**? YQ_a
From June 1st to November 30th is the **hurricane season**, but that doesn't mean there will always be...
Asked by robin c - 8 months ago - Other - Caribbean - 11 Answers - Resolved Questions

When is **hurricane season** in the **Caribbean**? YQ_b
is costa rica also affected? ... **Hurricane season** in the **Caribbean** (Atlantic) begins at the end of June and ...
Asked by shafan - 9 months ago - Other - Caribbean - 5 Answers - Resolved Questions

When is **hurricane season** in the **Caribbean**? YQ_c
Honeymoon for 7 days starting June 28...any thoughts/suggestions ... Where are you going? They can predict a "H" days in advance and if there is one out there, you...
Asked by - 8 months ago - Bahamas - 6 Answers - Resolved Questions

when is **hurricane season** in the **Caribbean** and Mexican coasts? YQ_d
... Same as the Atlantic (Gulf and **Caribbean** are considered...basin, in terms of how **hurricanes** are classified): June 1 - Nov. 30. The **season** peaks in early September...
Asked by JS - 1 year ago - Other - Travel - 4 Answers - Resolved Questions

What are the typical months of **hurricane season** in the **Caribbean**? YQ_e
... Officially, **hurricane season** stretches from June 1st to November 30th, and hits not only the **Caribbean** islands but also parts...
Asked by nipsy3 - 6 months ago - Other - Caribbean - 7 Answers - Resolved Questions

Figure 4: Top 5 results when searching "When is hurricane season in Caribbean?" on Yahoo! Answers.

The variants for the other two metrics, Precision and MAP (namely, **PREC_MAX, PREC_STRICT, PREC_RR, MAP_MAX, MAP_STRICT, MAP_RR**), are calculated similarly.

In summary, MRR_MAX (and PREC_MAX and MAP_MAX) represent the upper bound on Yahoo! Answers' accuracy (with the current retrieval and ranking algorithms) from the perspective of an intelligent searcher. This is an extremely strong family of baseline metrics, as it assumes an "oracle" that always makes the right decision to click on the question threads that contain the correct answer in the highest ranking position.

5. EXPERIMENTAL RESULTS

5.1 Learning Ranking Function

To learn the ranking function, we generate the training and testing data as follows: we randomly select 400 TREC queries from total 1250 TREC queries and collect all the related QA for these 400 queries. We use ten-fold cross validation to perform the training of the proposed ranking function using the algorithm introduced above. Ten-fold cross validation involves dividing the judged data set randomly into 10 equally-sized partitions, and performing 10 training/testing steps, where each step uses 9 partitions to train the ranking function and the remaining partition to test its effectiveness. Note that the following results were done on this smaller set train the ranking function - the main evaluation will be performed in the next section, over the remaining 850 TREC questions that were not used in training in any way.

Figure 5 reports the Precision at K for the hold-out validation data against each iteration of our learning algorithm. It can be clearly seen that Precision increases for the first

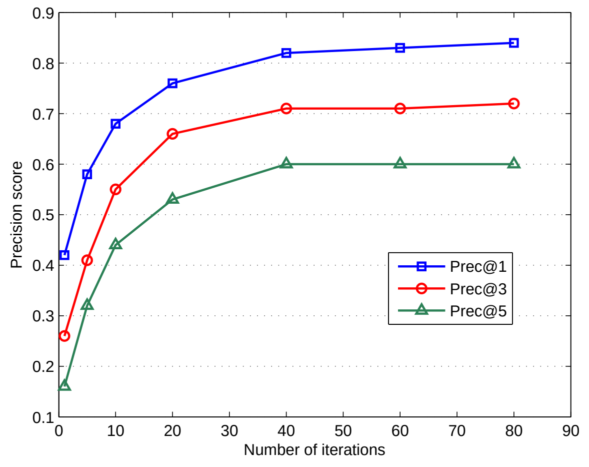


Figure 5: Precision at 1, 3, 5 for testing queries against GBrank iterations

60 iterations, after which the algorithm converges and additional iterations are not helpful.

5.2 Robustness to Noisy Labels

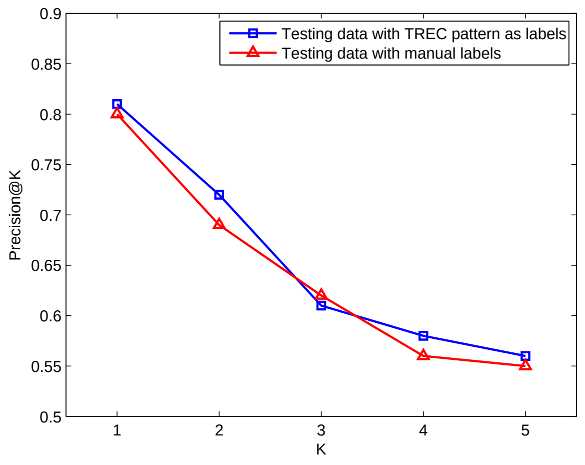


Figure 6: Precision at K for testing queries with manual labels and labels generated by TREC pattern

As mentioned in Section 4.1, the relevance judgments was obtained by matching an answer with TREC answer patterns. We have also found that 90% of items were given the same labels under both manual labeling and TREC pattern labeling and the remaining 10% of automatic labels were erroneous. To show that our learning framework is robust, we experiment with training on the noisy automatically generated labels, and testing on the smaller set of "gold standard" manually assigned relevance labels. For this experiment we used 50 queries (testing data) which have been labeled manually. Then, we randomly select the other 350 TREC queries (training data) and related questions and answers to train the ranking function. Figure 6 shows the Precision at K for testing data based on manual labels and TREC pattern labels respectively. While the accuracy when evaluating against manual labels is slightly lower than automatic labels, the differences are not substantial, which implies that our algorithm still generates a nearly-optimal model even when trained on noisy relevance labels. Furthermore, the high correspondence of automatic and manual label accuracy results validates our decision to use only automatic labels for the remaining experiments, to enable experiments on the larger scale.

5.3 Ablation Study on Feature Set

To gain a better understanding of the important features for this domain we perform an ablation study on our feature set to explore which features are significant to answers ranking.

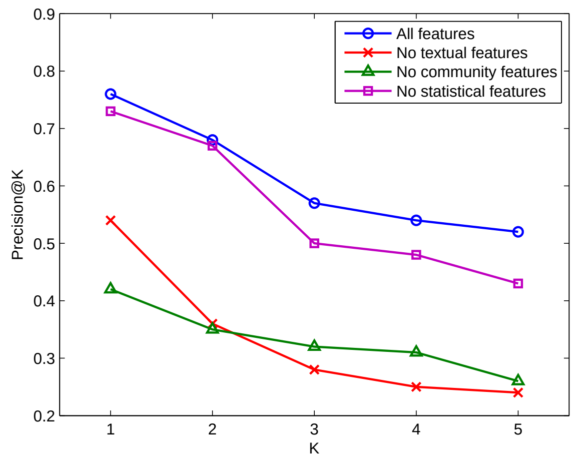


Figure 7: Precision at K for feature ablation study

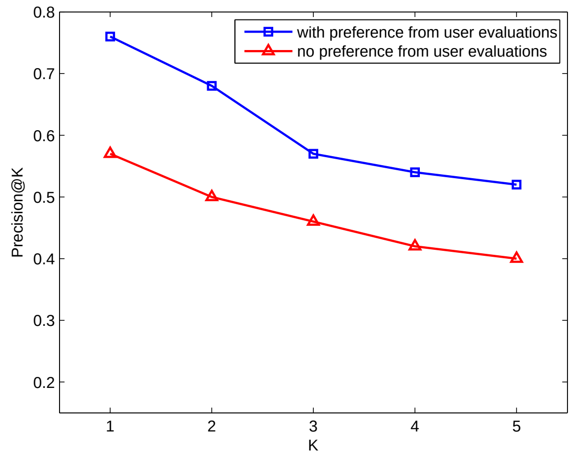


Figure 8: Precision at K without incorporating user evaluations as preference data

As shown in Table 1, there are three major categories of our feature set: textual features, community features and statistical features. Figure 7 reports the Precision at K

when learning ranking function with removing each category respectively. It is easy to see that removing both textual features and community features cause a significant decreasing on precision. While there is a slight reduction on precision when removing statistical features, it is clear that these features are not as important as the textual and community features. Interestingly, textual features are less important for Precision at 1. We hypothesize that for the top result it is more important for an answer to be chosen as "best" by the asker (one of the community features), than to have appropriate textual characteristics.

In addition, we also test the effectiveness of preference data from users evaluations. In order to test its effectiveness, we learn a ranking function without incorporating preference data from users evaluations. Figure 8 shows the Precision at K of this new ranking function. From this figure, we can see that users evaluations play a very important role in learning ranking function.

5.4 QA Retrieval

In this experiment, we train our ranking function on the whole training data (i.e., the 400 TREC queries from the previous experiments) and test on the remainder hold-out data of 850 TREC queries and associated community QA pairs.

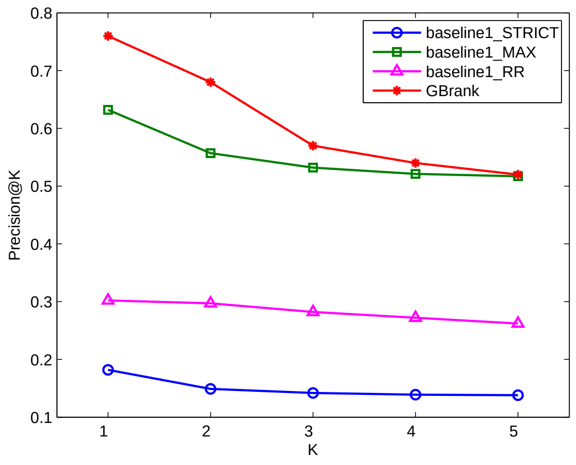


Figure 9: Precision at K for GBrank, baseline1_MAX, baseline1_RR and baseline1_STRICT for vary K

Figures 9 and 10 illustrate the Precision at K of GBrank compared with the two baseline methods. These figures show that the precision of two baseline methods are nearly the same, while GBrank out-perform both of them. In particular, the Precision at 1 of GBrank is 76%, compared to 63% precision at 1 exhibited by the _MAX baselines. We can also see that PREC_MAX performs better than PREC_RR and PREC_STRICT, and GBrank has the similar performance with PREC_MAX the values of K larger than 3.

In Table 2 and 3, we illustrate the MAP and MRR scores for two baseline methods as well as GBrank. From these two tables, it is clear that MAX can perform better than the other two metrics for baseline, but GBrank reaches much better performance than all the metrics for two baseline methods. In particular, GBrank achieves a gain of about 18% relative to the _MAX metrics.

To understand how GBRank can outperform an "oracle" baseline, consider that the ordering of answers within a question thread remains fixed (either by date – as the default, or by decreasing votes). In contrast, GBrank obtains a better ranking of answers within each question thread, as well as a global ranking of all answers. Then, improved ranking within each Yahoo questions thread contributes to the higher score than MRR_MAX. Overall, applied on Yahoo! Answers, our proposed framework achieves a significant improvement on the performance of QA retrieval over the Yahoo! Answers' default ranking and the supported optional votes-based ranking. In addition, from the experiment, we can find that our method is able to retrieve relevant answers at the top of results. In summary, we have shown that GBRank significantly outperforms extremely strong baselines, achieving precision at 1 of over 76% and MRR of over 0.78, which are high values even for traditional factoid QA.

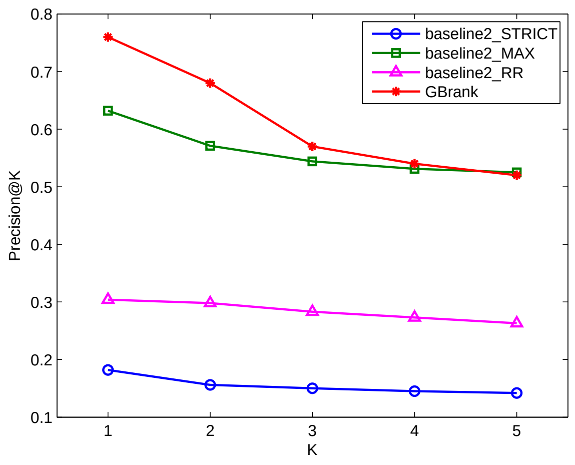


Figure 10: Precision at K for GBrank, baseline2_MAX, baseline2_RR and baseline2_STRICT for vary K

Table 2: Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP) for GBrank and other metrics for baseline1

	MRR	Gain	MAP	Gain
STRICT	0.213	0.569	0.043	0.422
ROUND ROBIN	0.401	0.381	0.145	0.310
MAX	0.662	0.120	0.441	0.024
GBrank	0.782	-	0.465	-

Table 3: Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP) for GBrank and other metrics for baseline2

	MRR	Gain	MAP	Gain
STRICT	0.214	0.568	0.045	0.420
ROUND ROBIN	0.403	0.379	0.159	0.306
MAX	0.664	0.118	0.443	0.022
GBrank	0.782	-	0.465	-

6. CONCLUSIONS AND FUTURE WORK

Community question answering is transforming the way people search for information. We have presented a robust, effective method for retrieving factual answers from community QA archives, and have demonstrated our method to be significantly more effective than the best possible accuracy a user can achieve when interacting with the current state-of-the-art question search on a major community QA site. Furthermore, our large scale experiments demonstrate that our method is robust to noise in the automatically generated training preference judgments. We have complemented our study with an analysis of the results to gain insight into the significant dimensions of fact retrieval from social media. In particular, we found that textual features and community features are crucial, and that user feedback, while noisy, provides sufficient relevance judgment to improve the learning of the ranking functions.

By significantly improving the accuracy of retrieving well-formed, factual answers, our work has the potential to transform how users interact with community QA sites; to improve the experience by reducing duplicate questions; and to better integrate question answering and search over QA archive with the mainstream web search results. In the future, we plan to extend this work beyond factoid question answering to complex questions and information needs. We also plan to extend our techniques to gracefully blend the results of social media search with organic web search results for the appropriate information needs, such as question answering. In summary, our work is a crucial component for factual information seeking in the increasingly important social media environment.

ACKNOWLEDGEMENTS We are grateful for the support of the Yahoo! Answers team to allow extensive usage of the Answers search API. We also thank Ke Zhou from Shanghai Jiaotong University for letting us use his gradient boosting code in implementing our ranking method.

7. REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of SIGIR*, 2006.

[2] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media with an application to community-based question answering. In *Proceedings of WSDM*, 2008.

[3] R. Baeza-Yates and B. Ribeiro-Neto. In *Modern Information Retrieval*, 1999.

[4] A. Berger. Statistical machine learning for information retrieval. In *Ph.D. Thesis, School of Computer Science, Carnegie Mellon Univ.*, 2001.

[5] E. Brill, S. Dumais, and M. Banko. An analysis of the askmsr question-answering system. In *Proceedings of EMNLP*, 2002.

[6] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of ICML*, 2005.

[7] R. Burke, K. Hammond, V. Kulyukin, S. Lytinen, N. Tomuro, and S. Schoenberg. Question answering from frequently asked question files: Experiences with the faq finder system. In *AI Magazine*, 1997.

[8] Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In *Journal of Machine Learning Research*, 2003.

[9] J. Friedman. Greedy function approximation: a gradient boosting machine. In *Ann. Statist.*, 2001.

[10] J. Jeon, W. Croft, and J. Lee. Finding similar questions in large question and answer archives. In *Proceedings of CIKM*, 2005.

[11] J. Jeon, W. Croft, J. Lee, and S. Park. A framework to predict the quality of answers with non-textual features. In *Proceedings of SIGIR*, 2006.

[12] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of KDD*, 2002.

[13] T. Joachims, L. Granka, B. Pang, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of SIGIR)*, 2005.

[14] P. Jurczyk and E. Agichtein. Discovering authorities in question answer communities using link analysis. In *Proc. of ACM Conference on Information and Knowledge Management (CIKM2007)*, 2007.

[15] D. Kelly and J. Teevan. Implicit feedback for inferring user preference: A bibliography. In *SIGIR Forum*, 2003.

[16] J. Ko, L. Si, and E. Nyberg. A probabilistic framework for answer selection in question answering. In *Proc. of NAACL HLT*, 2007.

[17] M. Lenz, A. Hubner, and M. Kunze. Question answering with textual cbr. In *Proc. of Third International Conference on Flexible Query Answering System*, 1998.

[18] J. Ponte and W. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR*, 1998.

[19] E. Sneiders. Automated faq answering: Continued experience with shallow language understanding. In *Proc. of the 1999 AAAI Fall Symposium on Question Answering System*, 1999.

[20] R. Soricut and E. Brill. Automatic question answering: Beyond the factoid. In *HLT-NAACL 2004: Main Proceedings*, 2004.

[21] Q. Su, D. Pavlov, J. Chow, and W. Baker. Internet-scale collection of human-reviewed data. In *Proc. of the 16th international conference on World Wide Web (WWW2007)*, 2007.

[22] E. M. Voorhees. Overview of the TREC 2003 question answering track. In *Text REtrieval Conference*, 2003.

[23] H. Zha, Z. Zheng, H. Fu, and G. Sun. Incorporating query difference for learning retrieval functions in world wide web search. In *Proceedings of CIKM*, 2006.

[24] J. Zhang, M. Ackerman, and L. Adamic. Expertise networks in online communities: Structure and algorithms. In *Proc. of International World Wide Web Conference WWW2007*, 2007.

[25] Z. Zheng, H. Zha, K. Chen, and G. Sun. A regression framework for learning ranking functions using relative relevance judgments. In *Proc. of SIGIR*, 2007.