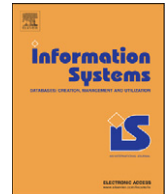




ELSEVIER

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

A framework for corroborating answers from multiple web sources

Minji Wu^{*}, Amélie Marian

Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854, United States

ARTICLE INFO

Article history:

Received 3 August 2010

Accepted 27 August 2010

Recommended by: D. Shasha

Keywords:

Corroboration

Mean reciprocal rank

Precision

TREC

Web search

ABSTRACT

Search engines are increasingly efficient at identifying the best sources for any given keyword query, and are often able to identify the answer within the sources. Unfortunately, many web sources are not trustworthy, because of erroneous, misleading, biased, or outdated information. In many cases, users are not satisfied with the results from any single source. In this paper, we propose a framework to aggregate query results from different sources in order to save users the hassle of individually checking query-related web sites to corroborate answers. To return the best answers to the users, we assign a score to each individual answer by taking into account the number, relevance and originality of the sources reporting the answer, as well as the prominence of the answer within the sources, and aggregate the scores of similar answers. We conducted extensive qualitative and quantitative experiments of our corroboration techniques on queries extracted from the TREC Question Answering track and from a log of real web search engine queries. Our results show that taking into account the quality of web pages and answers extracted from the pages in a corroborative way results in the identification of a correct answer for a majority of queries.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

Information available on the internet is abundant but often inaccurate. Users take for granted the availability of efficient search engines that will allow them to find the information they need. However, web search engines only point users to pertinent information available on the web, without vouching for its correctness.

Typical web search engines return a list of web pages (or sources) that matches a set of keywords input by users. Web search engines are increasingly efficient at identifying the best sources for any given keyword query, and are often able to identify the answer within the sources. Unfortunately, many web sources are not trustworthy, because of erroneous, misleading, biased, or

outdated information. With many web sources providing similar information on the Internet, users often have to rummage through a large number of different sites to both retrieve the information in which they are interested, and to ascertain that they are retrieving the correct information. In many cases, users are not satisfied with—or do not trust—the results from any single source, and prefer checking several sources for corroborating evidence, as illustrated in the following examples:

Example 1. Consider a user interested in buying a car, and considering a specific brand and make (e.g., Honda Civic). One of the criteria influencing the decision is the gas mileage of the car. However, the gas mileage information available on the Internet differs not only based on the year of the car, but also based on the source from which the information is extracted: the official manufacturer web site (up to 51 mpg in the Honda Civic example) has a different value than some other commercial web sites (40 mpg, *Autoweb.com*; 30/40 mpg, *Car.com*). None of these sources, all of which appear

^{*} Corresponding author. Tel.: +1 732 445 2001x9593;

fax: +1 732 445 0537.

E-mail addresses: minji-wu@cs.rutgers.edu (M. Wu), amelie@cs.rutgers.edu (A. Marian).

in the first page of results for the query “Honda Civic 2007 gas mileage” using MSN Search has the “perfect” answer for the query, but they all provide valuable information to the user.

Example 2. Consider a user who wants to find out who the first astronomer who orbited the earth was. Issuing the query “first orbited the Earth” to a commercial search engine returns a list of web sources that are relevant to the query but which provide different extracted answers as shown in Fig. 1. The correct answer, Yuri Gagarin, does not appear in the first search engine result; in fact, there are several results that does not contain any valid answer to the user’s information need. However, several answer extracted from the search engine result pages are potentially useful to the user as they provide correct answers to refinements of the user’s query: Valentina Tereshkova was the first woman who orbited the earth, and John Glenn was the first American to do so.

A naive approach to try to identify the correct answer would be to return the most frequent answer found among the search engine query result pages. While this method can efficiently eliminate outlier answers such as typos, it fails to consider the fact that answers extracted from different pages are rarely equally important in answering the query, as all pages are not equally trustworthy. In addition, it opens the gate to malignant behavior from spammers, who would be tempted to create multiple pages containing similarly erroneous information to boost the score of their chosen answer.

In this paper, we propose a framework to corroborate query results from different sources in order to save users the hassle of individually checking query-related web sites to corroborate answers. In addition to listing the possible query answers from different web sites, we rank the answers based on the number, relevance, and similarity of the web sources reporting them, as well as the prominence of the answers within the sources. The existence of several sources providing the same

Live Search | MSN | Windows Live | Hotmail

Live Search first orbited the earth

Web 1-10 of 154,000 results · Advanced
See also: [Images](#), [Video](#), [News](#), [Maps](#), [More](#) ▼

Featured Document: Friendship 7 Transcript ← **John Glenn**
The successful completion of Glenn's mission (he **orbited the Earth** three times) did much to restore American prestige worldwide. February 20, 1962
www.archives.gov/exhibits/featured_documents/friendship_7_transcript · [Cached page](#)

Yuri Gagarin - Wikipedia, the free encyclopedia ← **Yuri Gagarin**
On 12 April 1961, he became the **first** human in space and the **first** to orbit **the Earth**. He received medals from around the world for his pioneering tour in outer space.
en.wikipedia.org/wiki/Yuri_Gagarin · [Cached page](#)

Heliocentrism - Wikipedia, the free encyclopedia ← **No Answer Found**
... was available in the Tychonic system, in which the Sun **orbited the Earth**, while the planets **orbited** the Sun as in the Copernican model. The Jesuit astronomers in Rome were at **first** ...
en.wikipedia.org/wiki/Heliocentrism · [Cached page](#)

Flashback - 98.11.05 ← **John Glenn**
Atlantic Monthly articles on the space program and John Glenn's **first** flight orbiting **the earth** ... It has been almost four decades since John Glenn **first orbited the Earth**.
www.theatlantic.com/unbound/flashbks/glenn.htm

First Thai Observation Satellite To Be Orbiting In October ← **No Answer Found**
Bangkok (XNA) Jan 29, 2007 - Thailand is doing the final preparations for the launch of its **first earth** observation satellite called THEOS into orbit in October, Thai Science and ...
www.spacemart.com/reports/First_Thai_Observation_Satellite_To_Be_Orbited_In_October_999.ht... · [Cached page](#)

Valentina Tereshkova Biography ← **Valentina Tereshkova**
Valentina Tereshkova was the **first** woman in space, orbiting **the earth** forty-eight times in Vostok VI in 1963. She **orbited the Earth** for almost three days, showing that women have ...
www.notablebiographies.com/St-Tr/Tereshkova-Valentina.html · [Cached page](#)

Solar System Exploration: Missions: By Target: Earth: Past: Sputnik ← **No Answer Found**
... shot in the space race between the United States and the former Soviet Union. the basketball-sized spacecraft was the world's **first** artificial satellite. It **orbited the Earth** ...
solarsystem.nasa.gov/missions/profile.cfm?MCode=Sputnik · [Cached page](#)

Who was the First Astronaut? ← **Yuri Gagarin**
... the **first** human astronaut, Yuri Gagarin, on 12 April 1961, about three and a half years after the launch of Sputnik. Launched on board Vostok 1, Gagarin **orbited the Earth** just ...
www.wisegeek.com/who-was-the-first-astronaut.htm · [Cached page](#)

Fig. 1. Results for the query “first orbited the earth” using MSN search.

information is then viewed as corroborating evidence, increasing the quality of the corresponding information, as measured by a scoring function used to order answers. Our techniques are built on top of a standard web search engine query result, and use existing information extraction techniques to retrieve answers from web pages. Corroborating answers from web sources presents several challenges:

- The main challenge of answer corroboration is the design of a meaningful scoring function. The scoring function should aggregate similar answers and take into account a variety of parameters to identify the best answers.
- Accessing all the pages that match a given web search query to retrieve and compare answers would obviously be very inefficient. We need to select web sources that are most likely to contain the best answers.

We propose the following contributions to address these challenges:

- *Scoring of corroborated answers:* We propose a framework to score corroborated answers. Our approach considers several factors to score both the relevance of a page to the query and the importance of the query answer within the page. By combining these two factors we can assign a score to each individual answer based on how likely the answer is to be the correct answer. We then aggregate the score of similar answers. To the best of our knowledge, our techniques are the first to consider not only the frequency of the answers in the web search engine result, but also the relevance and originality of the pages reporting the answers, as well as the prominence of the answer within the page (Section 3). In particular, our web page relevance score is based on search engine rankings and modeled by Zipf's law, an intuition empirically validated using the user clicks from a search engine log.
- *Selecting the web sources from which to retrieve the answers:* By focusing on the pages that are most likely to contain good answers we are able to save on query processing time. This is related to work on top- k query processing and the Threshold Algorithm [16]; however, score bounds information, commonly used in top- k query processing, cannot be used in a corroborative framework. We propose a method to consider a prefix of the search engine query result for information extraction, dynamically deciding the size of this prefix based on the distribution of answers (Section 4). We experimentally evaluate the effect of the prefix size and show that our method is effective at reducing the number of pages necessary to corroborate answers.
- *Evaluating the quality of our proposed approach:* We conducted a novel extensive qualitative and quantitative experiments on queries selected from the TREC Question Answering Track [43] and from a log of MSN query searches. Our experimental results show that data corroboration significantly improves the quality

of answers (Section 5). We also show that for MSN queries, our corroborative answers correlate with user clicks in the MSN search log.

We report on related work in Section 6 and conclude in Section 7.

A preliminary version of this work has been published in [45]. The current paper extends our information extraction techniques to cover more queries, and improves on the simpler scoring model of [45] by considering modeling web page relevance using a Zipfian distribution (Section 3), an intuition we validate empirically. We also performed a new set of experiments on queries from the TREC Question Answering Track to test our approach, and compared our results with simpler frequency-based techniques (Section 5.2). In particular, we measured the quality of our results by comparing the corroborated answers with the TREC answers. Finally, we significantly expanded our evaluation on queries extracted from a real MSN query log (Section 5.3).

2. Extracting answers

Our corroboration system is built on top of an existing search engine. Given a user keyword query, our first step is to extract the candidate answers from the web pages returned by the search engine. The challenge is to efficiently retrieve, and identify, from these web pages the data that qualifies as an answer to the query. For instance, to retrieve answers for our Example 1 query, we need to identify gas mileage values that correspond to a 2007 Honda Civic from the search engine web page results.

In addition, it is often observed that the same answer may appear in different form in different sources. For example, we found two answers ("John Glenn" and "John H. Glenn") for our Example 2 query. While the text of these two answers is slightly different, it is highly probable they refer to the same answer. Our answer extraction system solves this problem by computing the cosine similarity score between answers and aggregating similar answers if their similarity score is above a certain threshold.

2.1. Answer extraction

Information extraction, the process of extracting relevant information from documents using text and structure, is a complex problem that has been the focus of many work in the natural language processing and machine learning communities [31], among others. Since the focus of our work is on the corroborative aggregation and scoring of answers, we opted to use regular expression techniques to extract answers from web pages. In addition, our current implementation works on queries for which the answers are a small piece of text, such as the ones illustrated in Examples 1 and 2.

It is relatively straightforward to identify numerical answers within web pages. For instance, mileage information will typically be of the form "x mpg," or "mpg of x," where x is a numerical value. Note that our extraction technique considers multiple units, and does the proper

conversions, for each numerical query (e.g., “feet” and “meters” for a length query).

It is more complicated to extract answers for a query that calls for a textual answer. State-of-art IE systems [1,11, 20–22,33,34] have used a lot of linguistic tools, namely syntactic parser, part-of-speech tagger, named entity tagger, WordNet.

Instead of employing a full featured information extraction system, existing works [35,23,37,47] in the information retrieval community have shown success by using a simplified answer extraction component. Radev et al. apply a part-of-speech tagger to phrases and computes the probability of phrase type to match the query category. Jijkoun and de Rejke pinpoint the answers for a query by looking at frequently asked questions (FAQ) archives. If a question in the FAQ archive is found to match the query, the non-question text block immediately following the question is identified as the answer for the query. The QA system in [37] retrieves answer sentences based on keyword matching. The QUALIFIER system in [47] performs answer selection by matching the expected answer type to the NLP results of the query and returns the named entity in the candidate sentence.

Our answer extraction is similar as the techniques used in [47]. Given a web page, we first apply a HTML parser to obtain the text content for answer extraction. We choose the Jericho HTML parser,¹ an effective Java open source HTML parser to obtain the plain text of the web page. We then use a tokenizer to tokenize and tag the plain text from the first step. The Stanford Name Entity Recognizer (NER) is a Java implementation of a Named Entity Recognizer which uses a linear chain conditional random field (CRF) sequence models. In particular, the Stanford NER system segments the plain text into sentences and words. After giving each word a tag, we apply extraction rules for each query by selecting the sentences that contain the extraction rule. The answer is then identified as the words from the sentences that match the expected answer type. The rules for each query are a set of automatically created texts that may appear around the answer. The rules are not query specific but rather are created based on the interrogative word to apply to different types of queries. We give a few examples of such rules for different queries in Table 1.

By using rules such as the ones shown in Table 1, we can extract answer for each query. As an example, consider the query: “Who is the speaker of the Lebanese Parliament”. We can create rules “is the speaker of the Lebanese parliament” and “the speaker of the Lebanese parliament is” to extract query answers.

While these rules cover a wide variety of scenarios, they are insufficient in some cases. Consider the following example:

“Nabih Berri reelected as the speaker of the Lebanese Parliament”

Clearly, “Nabih Berri” is an answer to the query, but the two previously created rules cannot extract it. Therefore, in

Table 1
Generating rules for queries.

Who is/was \$x	“is \$x”, “\$x is”
Who VBD \$x	“VBD \$x”, “\$x is/was VBN”
Where is \$x	“\$x is”
Where is \$x VBN	“\$x is VBN”
Where did/does \$x VB \$y	“\$x VBZ/VBD \$y”
Which \$x is/was \$y	“is/was \$y”, “\$y is/was”
Which \$x VBZ/VBD/VBN \$y	“\$y is/was VBN”, “VBZ/VBD/VBN \$y”
What \$x is/was \$y	“is/was \$y”, “\$y is/was”
What \$x does/did/have/has \$y VB/VBN	“\$y VBD/VBN \$x”

addition to the rules created as above, we introduce a set of *relaxed* rules for each query. The *relaxed* rules include the noun/verb phrase or part of the noun/verb phrase, possibly in combination with a specific adjective (e.g., “current”). We manually create a few specific adjectives that may appear around the extraction rule based on heuristics. For instance, the *relaxed* rules for the above query include “the speaker of the Lebanese Parliament”; “the speaker”; “the current speaker is”; “is the current speaker”.

Applying the Stanford NER recognizer will tag “Nabih Berri” as “PERSON”. Since the answer to this query is expected to be a PERSON entity and this sentence matches one of the rules of this query (i.e., “the current speaker is”), “Nabih Berri” is extracted as an answer to this query. Note that it is possible to extract multiple answers from the sentence which matches the extraction rule. We show how we assign scores to each of the extracted answers in Section 3.2.

2.2. Answer aggregation

After we have extracted answers from different sources, we need to combine similar answers. Due to the poor quality (spelling errors, typographical errors) of a large portion of online content, answers extracted from different sources bearing textural difference may actually point to the same information. A simple example is for numerical queries, different sources may present results using different unit. For instance, for our Example 1 query we may find answers “26 mpg (mile per gallon)” and “11 km/l (kilometers per liter)” from different sources which reflect the same gas mileage information for a Civic in city drive. Our solution to such cases is to convert all extracted answers into the same unit and combine answers that are similar in value. In our implementation, we consider answers that are within 5% difference in values as similar answers.

For factoid queries, we are dealing with problems of finding similar strings. Existing work [28,19,38] has been proposed using cosine similarity score to effectively handle spelling errors and rearrangement of words when comparing strings. The key idea is to obtain the *tf-idf* score vector for each string and compute cosine similarity score for each pair of strings. In particular, Koudas et al. leverage the semantic similarity if such information is available. In our corroboration system, we apply the *tf*-based cosine similarity method to detect similar answers. The cosine

¹ <http://jerichohtml.sourceforge.net/doc/index.html>

similarity score of two answers is computed using the word frequency vector of the two answers. Given two tokenized answers, we first need to obtain the word frequency vectors for the two answers W_1 and W_2 , the cosine score is then computed as the dot product of the two vectors divided by the square root of the product of the vector dot products of each score vector with itself. For example, assuming we have extracted answers “John Glenn” and “John H. Glenn” for our Example 2 query, the word frequency vectors W_1 and W_2 for the two answers are (1, 0, 1) and (1, 1, 1). The cosine similarity score is therefore computed as the dot product of W_1 and W_2 (which is 2) divided by the square root of product of each vector with itself (which is $\sqrt{2} \cdot \sqrt{3}$). The cosine similarity score (0.82) is then compared against a user-defined threshold to test if the two strings can be classified as similar answers. In our implementation, we use the threshold of 0.8 and we recognize “John Glenn” and “John H. Glenn” as similar answers.

3. Scoring answers

Once we have extracted answers from the web pages returned by the search engine, we need to identify the best answer to the query. For this purpose, we need to assign scores to each extracted answer. We then aggregate the score of similar answers to identify the best corroborative answer.

In order to assign a score to each extracted answer, we first assign to each web page a score that represents the likelihood that an answer found within the page is the correct answer. Then, for each answer found within a web page, we assign a score that represents the probability that this answer is the correct answer within the page. Finally, scores of similar answers are aggregated to provide a corroborative answer score.

The score of an answer x extracted from a given web page p is the product of the score of the page ($P(p)$) and the score of the answer within the page ($P(x|p)$).

$$P(x,p) = P(p) \cdot P(x|p) \quad (1)$$

In the rest of this section, we detail our scoring approach. We explore several scoring components that can be used in conjunction, or separately. We will discuss the impact of each of the components, and evaluate them experimentally in Section 5.2.2. In Section 3.1, we propose a scoring method of individual web pages. We then propose techniques to estimate the score of an answers within a page in Section 3.2. The corroborative score of an answer among all search engine query result pages, and taking into account each of our proposed component, is given in Section 3.3.

3.1. Scoring web pages

The score of an answer depends on the quality of the web page from which the answer is extracted. We consider two factors in measuring the quality of a web pages: the relevance of the page (Section 3.1.1), and the originality of the page (Section 3.1.2).

3.1.1. Web page relevance

Search engines rank web pages according to (among other factors) the relevance between the page and the query keywords. An intuitive method for web page scoring is to use the page rank score of web pages as given by the search engine. Unfortunately the search engine does not provide its internal score along with the ranking. The *tf-idf* score has been widely used in the information retrieval community to compute the relevance of a document with respect to a keyword query, but our system is built on top of a search engine; as such we do not have access to indexes of the whole web. Therefore we use the individual ranks of web pages in the search engine result list as a measure of relevance.

As we traverse the search engine query results, the quality of the match decreases. We consider that a page ranked highly by the search engine is more likely to provide good information than a page with a lower rank. Therefore an answer found within a higher ranked page will have a higher probability than one found within the lower ranked page to be the correct answer. To model this decrease of the relevance of web pages as we go down the search engine result we use Zipf's law distribution function [50], commonly used in natural language processing to model the distribution of words in languages where very common words have very high frequencies.

Previous work [7] refers to the commonness of Zipf-like distribution in web access patterns. For instance, there is evidence that the popularity of web pages follows a Zipf distribution. In this work, we are interested in the importance of web pages as we go down a search engine query results. We investigated the click patterns of a query log of about 15 million MSN queries. Fig. 2 shows the log-log plot for the distribution of user clicks depending on the position of the page in the search engine query result, for the first 10 results and for all queries in the log. We only report on the first 10 results here since the number of user clicks shows a sharp decrease after the 10th result (and similarly after the 20th, 30th, etc.) as search engines return results 10 at a time and users are reluctant to go beyond the first page of result.

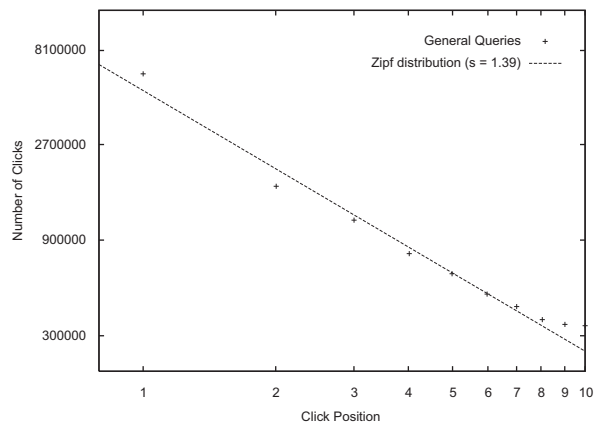


Fig. 2. The log-log plot for the number of user clicks as a function of the position of the page in the search engine query result.

We used curve-fitting techniques,² based on a linear regression in the log-log plot to approximate the distribution of the user clicks per position. Our results show that the distribution can be approximated to a power law distribution, with an exponent parameter (s) of 1.39, and constant close to 1.

These results show that the Zipf distribution is adequate to approximate the decrease in page relevance. We then define the relevance score of a page as

$$P(p) = \frac{1/r(p)^s}{\sum_{i=1}^N 1/i^s} \quad (2)$$

where N is the total number of pages considered (i.e., the estimated size of the search engine query result), and $r(p)$ is the rank of page p in the search engine query result. We normalize the score, so that the sum of all page scores are summed to 1.

The s exponent parameter has an impact on the slope of the score distribution. It quantifies how quickly the quality of answers degrades when we traverse the search engine result list. In our system, varying s will have an impact on both the quality of answers and the number of pages retrieved to identify the best answer. We explore the effect of s experimentally in Section 5.2.

3.1.2. Web page originality

In addition to the web page relevance, we take into account the originality of the pages to reduce the effect of similar information coming from sources in the same domain, or sources that seem to be mirror (or copy-paste) information from each other. These pages tend to show some strong correlation as they reflect information coming from the same real-world sources. As an example, many web pages directly copy material from high-profile web sources such as *Wikipedia*. Errors in the original web site are therefore propagated to independent web pages. Our motivation for using corroboration is to confirm evidence from several autonomous sources. Considering sources that mirror each other as completely separate sources for corroboration would then artificially increase the weight of the original real-world sources (such as *Wikipedia*), and would open the door to some possible malignant behavior from content providers in order to boost the ranking of a given answer. However, discarding the redundant source is not a good option either, as its existence leads some more credence to the information, although not as much as if it were a source containing original information.

Since we aim at corroborating answers from different web pages, duplicated pages which tend to have a similar content should not have as much weight in the corroboration as pages that contain original content. However, they should still be taken into account as corroborative evidence, as there is a possibility they reflect similar information coming from independent sources. Our solution is to dampen the score of a page each time a duplication is detected. Our implementation detects suspected copy-paste as duplicated text around answers.

Detecting near duplicate web pages is a non-trivial task and has been the focus of a large body of research. Often, two pages sharing the same core content may be classified as independent sources using byte-wise comparison due to different framing, advertisements, and navigational banners. Among those existing works, the SpotSigs technique proposed in [39] proves to be an effective way to detect such near duplicate pages. The key idea in SpotSigs technique is to create a *robust* document signature with a natural ability to filter out noisy components of Web pages. In our corroboration system, we implemented the SpotSigs technique to detect copy/paste pages. Our current implementation is Boolean, a page is a copy or it is not, but we could easily extend our system to use more complex copy detection tools such as [9,6] to identify different degrees of duplication, and use this information to provide a finer granularity of web page originality scores.

We introduce a parameter β to quantify the redundancy of duplicated pages. When traversing a search engine result list, we check whether each page contains original content, or whether it is similar to a page with higher rank. If for a page p there exists $d_m(p)$ higher ranked pages sharing the same domain and $d_c(p)$ higher ranked pages generated from copy/paste information, we adjust the relevance of the page as follows:

$$P(p) = \frac{1/r(p)^s}{\sum_{i=1}^N 1/i^s} \cdot (1-\beta)^{d_m(p)+d_c(p)} \quad (3)$$

The first occurrence of a page from any given domain (or from duplicated web pages) will therefore be assigned its full score, only subsequent similar pages will have their scores dampened. This ensures that we are taking into account all original information, and limits the weight of redundant information in the scoring.

3.2. Scoring answers within web pages

It is common for several, possibly different, answers to be extracted from a single web page. This can be due to some error or uncertainty in the answer extraction process, or to the fact that the web page does contain several answers. To identify the best answer, we are then faced with the challenge of scoring these multiple answers. If only one answer x is extracted from a page p , it gets a score of 1 for that page ($P(x|p) = 1$). A simple way to score multiple answers stemming from the same page would be to assign each of them a score of $1/N(p)$, where $N(p)$ is the number of answers in the page. One problem of the above method is that all the answers extracted from the same page are rarely equally helpful in answering queries. Consider the following text from which we can extract two answers from a single web page for the query in Example 2 “first orbited the earth”.

Example 3. “Now you could have asked, even at the time, what was so special, so magical, about *John Glenn*, since a year before him a Russian, one *Yuri Gagarin*, was the first human to orbit the Earth in space and four months after him another Russian did it 17 times”.

² <http://www.fast.u-psud.fr/ezyfit/>

By applying the answer extraction techniques (Section 2) we could extract two answers, namely, “Yuri Gagarin” and “John Glenn”, which are underlined in the above example. Unfortunately, due to the limitation of the simple information extraction techniques we are using, it is difficult to figure out which one the correct answer is to the query. Our solution is to consider the *prominence* of each answer extracted within the page. We define the *prominence* $M(x,p)$ of an answer x from page p as the inverse of the distance $dis(x)$ between this answer and the extraction rule (i.e., “the first human to orbit the earth” in this case).

$$M(x,p) = \frac{1}{dis_{min}(x)} \quad (4)$$

We compute $dis(x)$ as the number of tokens plus 1 between the extraction rule and answer x , with a minimum distance of 1. As mentioned in Section 2.1, it is possible for multiple answers to be extracted via an extraction rule. The closer an answer is to the extraction rule, the more prominent it should be to answer the query. This approach is based on the assumption that in most cases, relevant answers will be close to the extraction rule in the web pages. We compute the score of the answer within the page as its normalized prominence. As for example 3, we have $dis(\text{“Yuri Gagarin”}) = 2$ and $dis(\text{“John Glenn”}) = 12$. Therefore the prominence scores for the two answers are 0.86 and 0.14, respectively. Formally, if an answer x is extracted from page p out of $N(p)$ answers, we define the score of answer x to be as follows:

$$P(x|p) = \frac{M(x,p)}{\sum_{i=1}^{N(p)} M(x_i,p)} \quad (5)$$

Our simple method to compute prominence scores leads to improvements in answer quality (Section 5.2.1). It is possible that the use of more refined information extraction techniques [31,13] that return answers with an associated confidence score from which we could derive our prominence score would result in further improvements.

Finally, some web pages may provide several answers; as for Example 1 the `CAR.COM` web site gives two answers for our query: 30 mpg (city), and 40 mpg (highway). Our current implementation penalizes answers that are not unique in their sources. In some cases, as in the city/highway example, multiple answers in a given web page may be due to different context. We plan to add context to our scoring approach in the future, possibly enabling our techniques to output different answers for different contexts.

3.3. Corroborating answers

Taking into account the scores of web pages as well as the scores of answers within web pages, we can assign the score of an answer x from a page p as

$$P(x,p) = \frac{1/r(p)^s}{\sum_{i=1}^N 1/i^s} \cdot (1-\beta)^{d_m(p)+d_c(p)} \cdot \frac{M(x,p)}{\sum_{i=1}^{N(p)} M(x_i,p)} \quad (6)$$

The *frequency* of the answers in the set of pages is considered in our corroboration approach. Intuitively, an answer that appears in 10 pages is more likely to be the correct answer than an answer that appears in one page, unless those 10 pages have very low scores. Formally, if $P(x,p_i)$ is the score of answer x from page p_i , the corroborative score of answer x is given by

$$P(x) = \sum_{i=1}^n P(x,p_i) \quad (7)$$

where n is the number of pages we consider from the search engine query result.

4. Retrieving pages

Finding and computing the scores of answers are not the only challenges we face when corroborating answers from search engine query results; another challenge is to select the set of result pages from which we extract information. As we go from higher ranked to lower ranked pages, we have to decide how deep we should go in the search engine query result. Accessing all the pages that match a given web search query to retrieve and compare answers would obviously be impractical and inefficient. In addition, lower ranked pages, which show little correlation with the query tend to give “noise” instead of useful information. Work on top- k query processing algorithms have focused on adaptively reducing the amount of processing done by query evaluation techniques by ignoring data that would not be useful to identify the best answers to a query [16,30]. However, these techniques cannot be directly applied to our scenario as they rely on query models where the score upper bounds of query results are known, and use this information during query processing. In contrast, in our corroborative model, the score of an answer can potentially grow every time a new page is retrieved.

We adapt ideas from work on top- k query processing to the answer corroboration problem by adaptively selecting a subset of search results from which to extract answers, based on the current scores of the retrieved answers. As was suggested in [27], we focus on estimating the maximum possible score increase that an answer could receive from unretrieved pages, in the absence of score upper bound information. Succinctly, we process web pages in the search engine result order, and stop retrieving new pages when the score of newly discovered answers would not be high enough to impact the overall corroborative answer score ranking.

We use our page relevance score (Section 3.1.1) to decide when to stop retrieving new pages from the search engine result. As we traverse the search engine result list, the relevance score of new pages we encounter decreases following a Zipf distribution. We stop retrieving new pages when the sum of the relevance scores of the unretrieved pages is too small for any answer extracted from new pages to cause any significant change to the corroborative answer list.

The maximum possible score increase is defined as the sum of all scores of unretrieved pages. Based on Eq. (2),

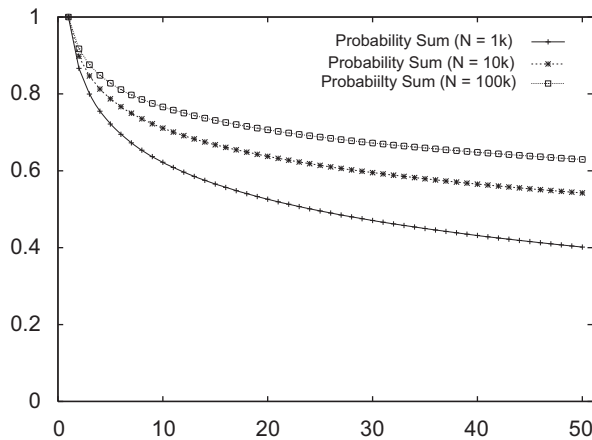


Fig. 3. Decrease in value of I_{Max} as we traverse the search engine query result.

this score I_{Max} is defined as

$$I_{Max} = 1 - \sum_{i=1}^r P(p_i) \quad (8)$$

where r is the rank of the last retrieved page, and p_i is the page retrieved at rank i . The value of the maximum possible score increase I_{Max} constantly decreases as we traverse the search engine query result.

During corroboration processing, we maintain a threshold variable T , which represents the value required to cause significant changes to the current result list. Our current implementation considers T to be the current difference in score between the top-1 answer and the top-2 answer. That is, we stop retrieving new pages when the current top-1 answer cannot change, i.e., when $T \geq I_{Max}$.

Although we dynamically retrieve pages and stop as soon as the remaining unretrieved pages will not make significant changes to the current answer list, we may still end up retrieving a large number of pages. This is due to our Zipf's law model where a large number of lower ranked pages may add up to a high score, as shown in Fig. 3, which shows the decreasing value of I_{Max} as we traverse the search engine result, up to page 50, for various values of N (the estimated total number of pages in the query result).

To address this, we limit the maximum number of pages our system will retrieve. This limit can be user-defined. By limiting the maximum number of pages retrieved, we are ignoring a subset of the web pages as well as the corresponding score increase they may bring to the corroborated answer scores. This unused potential score increase depends both on the maximum number of pages retrieved, and on the expected total number of page considered (as estimated by the search engine query result size). The normalization factor of Eq. (2) is adjusted to consider the maximum number of pages $maxPage$, as follows:

$$P(p) = \frac{1/r(p)^s}{\sum_{i=1}^{maxPage} 1/i^s} \quad (9)$$

We use deterministic bound information to decide when to stop retrieving new pages. An interesting direction to investigate would be the use of probabilistic bound information instead, in the manner of [40]. We are planning to investigate this approach in future work.

5. Evaluation

In this section, we present our in-depth experimental evaluation of our corroboration approach for web searches. We describe our experimental setup in Section 5.1. Section 5.2 focuses on queries taken from the TREC Question Answering Track and derives the best setting of s , $maxPage$ and β . Results for real-user queries from a MSN query log are given in Section 5.3.

5.1. Experiment setup

Our system is implemented using Java SDK 1.5 and built on top of the MSN search SDK. We use MSN search as our backbone search engine. We ran our experiment on machines running Fedora 6 with 4 2.8 G CPU and 2G RAM.

5.1.1. Evaluation queries

We reported preliminary experiment results on numerical queries in [45]. In this paper, we extend our experimental evaluation to a broader range of queries, including queries with textual answers. We consider queries from two sources: the TREC Question Answering Track and a MSN Live Search query log that contains real-user queries that were evaluated by MSN in the Spring of 2006.

TREC question answering track: We extracted 42 numerical queries and 100 factoid queries from the TREC Question Answering Tracks from TREC-8 (1999) to TREC 2006. We use keywords to extract numerical (e.g., “length of”, “height of”) and factoid queries (e.g., “who”, “where”) from the TREC QA track. Example numerical queries include: “diameter of the earth”, “average body temperature”, “length of Columbia River”, “height of the tallest redwood”. Example factoid queries include: “Who is the Speaker of the Lebanese Parliament”, “Who is the manager of Manchester United”, “Where is Merrill Lynch headquartered”, “Where was Hitchcock born”.

We use the TREC original answers to evaluate the quality of our corroborated answers. Note, however, that we are running our queries on top of a web search engine and therefore use the web as a data corpus rather than the TREC documents. Note also for some TREC queries, more than one answers are recognized as the correct answer (for example, for the query “Where did Jay-Z grow up”, both “brooklyn” and “New York” are correct answers according to TREC judgement). For such queries, we evaluate the corroborated answer with the highest ranked correct answer.

MSN Live Search query log: We selected 38 numerical queries and 100 factoid queries from the MSN search log using similar patterns as the TREC QA track. We manually pick queries to filter out non-factoid queries (for example, “Who wants to be a millionaire”) and only considered

queries which yielded at least one user click on the search engine query result.

5.1.2. Evaluation measures

We report on the following evaluation measures.

- **Percentage of queries correctly answered (PerCorrect):** For the TREC dataset, we compare our top corroborated answers with the original TREC answers. We derive the percentage of queries with correct answers at different rank of our corroborated answer list (top-1 to top-5).
- **Mean reciprocal rank (MRR):** The mean reciprocal rank of the first correct answer (MRR) is generally used for evaluating the TREC Question Answering Tasks. If the query does not have a correct answer in our top-5 corroborated answer list, its reciprocal rank is set to 0. We report the MRR values for experiments over the TREC queries.
- **Similarity between user clicks and corroborated answer:** Unlike TREC queries, queries extracted from the MSN query log do not come with an original answer. To evaluate the quality of our corroborated answers, we therefore compare pages that generate top corroborated answers with user clicks.
- **Time cost:** We report the time needed to return the corroborated answer list. This time cost is divided into retrieval time, which is the time spent accessing the web pages, and corroboration time, which is the time spent by our system for answer extraction and scoring.
- **Number of pages retrieved:** As discussed in Section 4, we do not retrieve every page from the search result but dynamically stop when we have identified the top-1 corroborated answer. We report on the actual number of web pages needed to reach an answer.

5.2. TREC queries

We now report our results on queries from the TREC QA Track. In this section, we first discuss the quality of our answers (Section 5.2.1). In particular, we show the benefit

of each individual scoring components on the answer quality (Section 5.2.2). We then compare the performance between our approach and previous question answering techniques (Section 5.2.3). We will show the number of web pages needed to return an answer (Section 5.2.4), and the time cost of our techniques (Section 5.2.5).

5.2.1. Answer quality

We first evaluate the quality of our corroborated answers by analyzing the percentage of correct answers (compared to the TREC-supplied original answers) for the 142 queries we extracted from TREC. Note that as shown in Eq. (6) there are three parameters which may affect the answer score: s , β and $maxPage$. In the following we evaluate the effect of each of these parameters.

We first fix β to 0.25 and $maxPage$ to 50 and test the effect of s , which is the parameter in Zipf's distribution. Fig. 4 shows the PerCorrect values for our top-1 to top-5 corroborated answers for different values of the s parameter of our Zipf corroboration scoring method (CORROB) of Section 3. Overall, an s value of 1 provides the best results at top-1. Interestingly, the distribution of user click per result position for the subset of the MSN queries that only considers queries expecting a numerical answer has a slope which is not as steep as the one for general queries (Fig. 2) as these queries yield more clicks, which end up increasing the probability of clicks for positions higher than 1. Using the same curve-fitting techniques as we did in Section 3.1.1 we can approximate the click distribution of these numerical query to a power law distribution with an exponent parameter (s) of 1.01 (Fig. 7(a)). In addition, the curve-fitting yields an exponent parameter of 0.92 for the click distribution of the factoid queries (Fig. 7(b)), which have the best PerCorrect and MRR values with a s value of 0.8 and 1.0. A s value of 1.0 yields a slightly better PerCorrect value at top-2 answer but slightly worse PerCorrect value at top-5 answer compared with a s value of 0.8. Both of these results validate our choice of Zipf distribution to model the originality of pages.

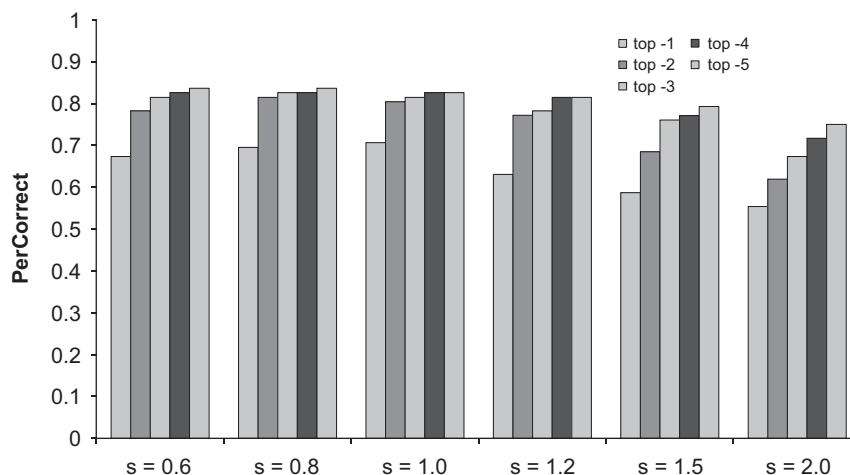


Fig. 4. Impact of parameter s on PerCorrect for TREC queries for CORROB.

As we increase s , the quality of our corroborated answers drops since top search engine result pages are given the most relevance weight, and fewer pages are considered in the corroboration. Inversely, for lower values of s , many pages are considered for corroboration, but their relevance scores tend to be similar and answers from high-ranked search engine query result pages are not given more weight.

Figs. 5 and 6 plot the PerCorrect values for numerical and factoid queries, respectively. As shown, both results are consistent with the overall PerCorrect values.

Table 2 reports the MRR value for the CORROB method presented in this paper. We list the MRR values for all 142 TREC queries and for the two separate types of queries in each of the three columns. We obtain a MRR score of 0.767 for the CORROB method with an s value of 1.0. This means that on average, the correct TREC answer is found within the two best corroborated answers. For comparison, the web question answering system presented in [14] has a best MRR score of 0.507; however, they report

results on a different subset of TREC queries that contains a broad variety of queries, which are possibly harder to answer. In the rest of our experiments, we will use $s=1$ as our default parameter value.

Another factor that affects the performance of our CORROB method (Eq. (3)) is the parameter β that we use to quantify the decrease of duplicate page score. Fig. 8 plots the PerCorrect values for the top-1 to top-5 corroborated answers, and Table 3 shows the MRR values of the CORROB method with a β value of 0, 0.25, 0.5, 0.75 and 1 and with a fixed s value of 1.0 and $maxPage$ value of 50. We discussed in Section 3.1.2 that duplicated pages should not have as much weight as page that contains original information but still need to be taken into consideration. The results shown in Fig. 8 and Table 3 confirm our claim. As shown, both PerCorrect and MRR values show improvements compared with the case that no originality is considered ($\beta = 0$). However, if we completely remove the effect of duplicated pages by setting $\beta = 1$, the answer quality is not as good as the case for which no originality

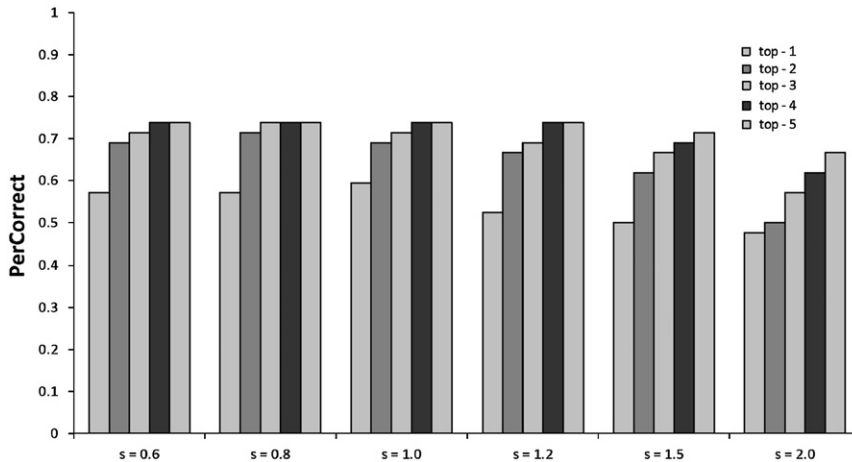


Fig. 5. Impact of parameter s on PerCorrect for numerical queries for CORROB.

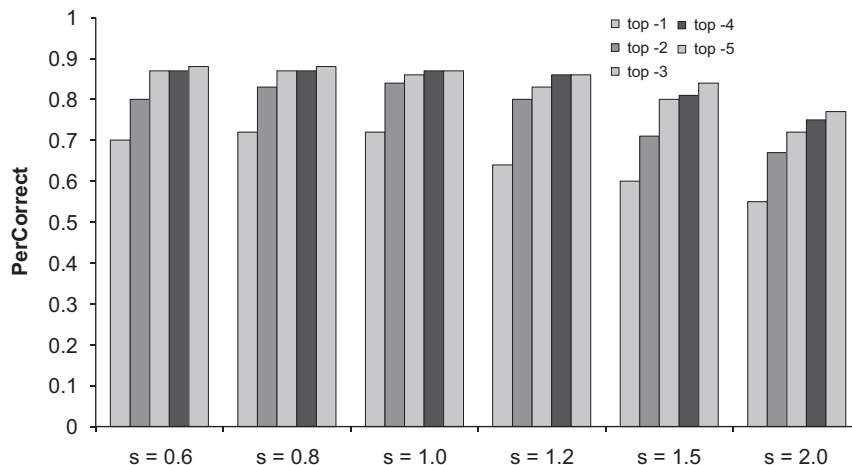


Fig. 6. Impact of parameter s on PerCorrect for factoid queries for CORROB.

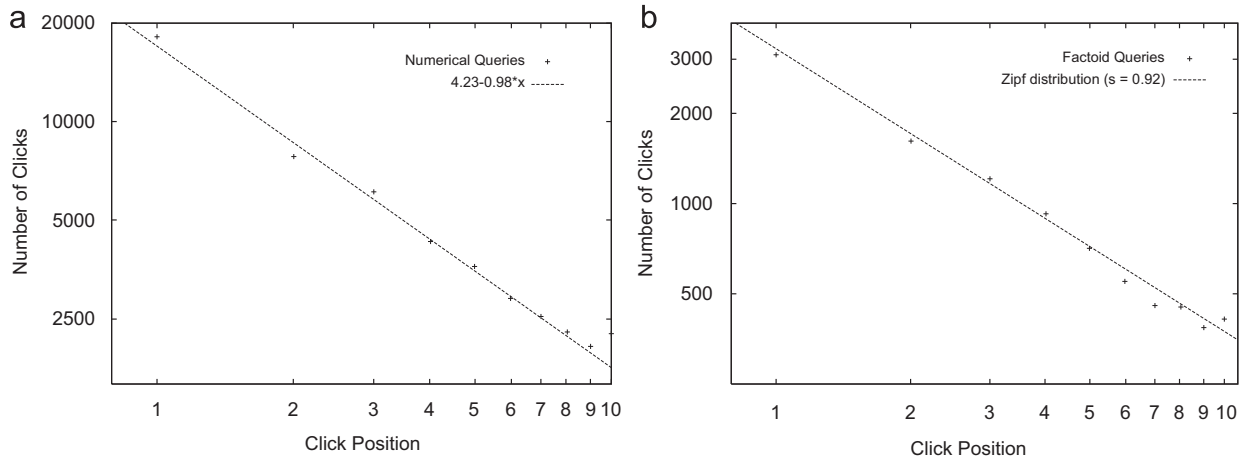


Fig. 7. The log–log plot for the number of user clicks as a function of the position of the page in the search engine result.

Table 2

Answer quality for TREC queries.

	Combined	Numerical	Factoid
CORROB ($s=0.6$)	0.752	0.645	0.797
CORROB ($s=0.8$)	0.765	0.651	0.813
CORROB ($s=1.0$)	0.767	0.657	0.813
CORROB ($s=1.2$)	0.704	0.615	0.741
CORROB ($s=1.5$)	0.664	0.586	0.697
CORROB ($s=2.0$)	0.610	0.533	0.642

is considered. Overall, the CORROB method has the best answer quality when we set $\beta = 0.5$. In the following experiment, we set 0.5 as the default value for β .

As discussed in Section 4, limiting the number of pages retrieved using the *maxPage* parameter may speed up query evaluation, but could affect answer quality as fewer web pages will be used for corroboration. Fig. 9 shows the impact of different *maxPage* value (from 20 to 90) on the percentage of correct answers. The answer quality increases as *maxPage* goes from 20 to 50, but remains the same when more pages are considered. As expected, considering too few pages for corroboration decreases answer quality. Our approach retrieves fewer than 50 pages (on average 34 pages for $s = 1$, shown in Fig. 12) to return a corroborated answer, therefore increasing *maxPage* above 50 has little impact on the quality of the corroborated answers. In addition, high values of *maxPage* lead to retrieving more pages, resulting in higher query processing times. Therefore, in the rest of our experiments, we set the value of *maxPage* to 50.

Our CORROB technique finds the correct answers within the top-5 corroborated answers for 85.2% of the 142 queries we extracted from TREC. For 21 of the TREC queries, we were not able to identify the correct answer within the top-5 corroborated answers. An in-depth analysis of these 21 queries shows that four of them are multi-answer queries (e.g., “width of Atlantic Ocean”) for which our top-5 corroborated answer list contain at least one correct answer (but not the TREC original answer); we

were not able to extract web answers for 10 of the queries (e.g., “the first director of the World Food Program”); we answered incorrectly 5 queries; and finally, 2 of the queries have wrong (or possibly outdated) answers within TREC: “Lifetime of hermit crabs,” which has a TREC answer of 70 years, and a correct answer of 30 years, and “highest recorded temperature in San Antonio, TX,” with a TREC answer of 107F and a correct answer of 111F. Out of the 142 queries, 22 of them were time sensitive (for instance, “who is the manager of Manchester United”). However, at the time of our experiments, the correct current answers were the same as the ones provided by the TREC corpus.

5.2.2. Impact of scoring components

In this section, we show the benefit of each individual scoring components we present in Section 3. In the following comparison, we denote the baseline BASE as the scoring approach in which the score of pages are the same, and the score of each answer is the score of the page divided by the number of answers within the page. We use ZIPF to denote the method in which only Zipf’s page relevance is considered (Section 3.1.1). We use ORIG to denote the baseline approach with the addition of web page originality (Section 3.1.2). We use PRO to denote the baseline approach with the addition of answer prominence within web page (Section 3.2). The CORROB approach, which combines ZIPF, ORIG and PRO, refers to the comprehensive scoring approach, as described in Eq. (6).

Fig. 10 shows the PerCorrect values for our top-1 to top-5 corroborated answers, and Table 4 reports the MRR values for different combinations of the scoring components. In most cases, the answer quality improves as we incorporate each of the scoring component, the ZIPF component provides the best improvement. The only exception is when adding the ORIG component, the PerCorrect value at top-1 slightly decreases, but combined with ZIPF components, the originality provides a small increase in answer quality. The CORROB scoring approach,

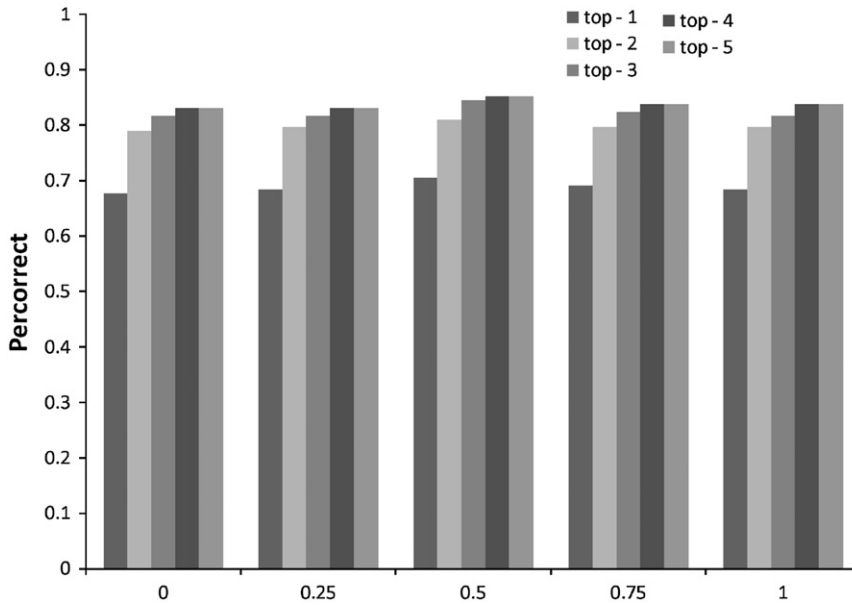


Fig. 8. Impact of β on PerCorrect for TREC queries.

Table 3

Impact of β on MRR for TREC queries.

β	MRR
0	0.766
0.25	0.767
0.50	0.772
0.75	0.756
1.0	0.752

which combines all three components, has the best result, with an MRR value of 0.772, outperforming the *BASE* case by 6.9%.

5.2.3. Comparison with existing question answering techniques

In this section, we compare our approach with existing question answering techniques. Previous works in question answering have been using answer frequency as the evidence of answer quality [29,14,48]. Summarizing previous work as frequency-based approach is a reasonable simplification. We could not exactly reproduce the actual question answering techniques of these works because we do not have access to the detail of their implementations, such as their information extraction and query rewriting techniques. Our corroboration approach could be used in conjunction with these tools to further improve answer quality.

In a frequency-based approach, the score of an answer is a function of the frequency of the answer among all the pages. In particular, we implement two types of frequency-based approaches: page-frequency (*P-FREQ*) and answer-frequency (*A-FREQ*). In *P-FREQ*, the score of an answer is based on the number of pages from which this answer was extracted. In *A-FREQ*, the score of an

answer is based on the number of time the answer was extracted from all pages.

In addition, we also implemented *TOP-PAGE* approach, in which we only extract answers from the first page and rank answers based on their frequency (*A-FREQ*). We tested the performance of *TOP-PAGE* simply based on the observation that most users only look into the first page returned from the search engine.

Fig. 11 and Table 5 show the PerCorrect and MRR values of the two frequency-based approaches, the *TOP-PAGE* approach, the *BASE* approach, the *ALPHA* method [45], and our *CORROB* approach. As shown, *P-FREQ* and *A-FREQ* achieve a PerCorrect value of 0.57 at the top-1 corroborated answer and 0.77 at top-5 corroborated answers. In addition, *P-FREQ* and *A-FREQ* have MRR values of 0.663 and 0.664, respectively, both of which are smaller than the *BASE* approach, showing that even a simple corroboration-based approach outperforms the frequency-based techniques dramatically. We compute the statistical significance of our results using the one-tailed Wilcoxon Signed-Rank test [44]. The difference between *CORROB* and *A-FREQ* and between *CORROB* and *P-FREQ* are both statistically significant with a *p-value* of 0.003. These results confirm that a corroboration-based approach outperforms frequency-based approaches. In addition, the advantage between the *CORROB* method and the *BASE* approach is statistically significant with a *p-value* of 0.034. The *TOP-PAGE* approach, performs surprisingly bad compared with all other techniques, with an MRR value of 0.36. This suggests that looking into only the first page from the search engine result is not sufficient for the users to get the correct answer in most cases.

We proposed the corroborative *ALPHA* method in [45]. *ALPHA* uses a simpler scoring strategy for the page relevance, dropping the score of a page *p* based on a parameter

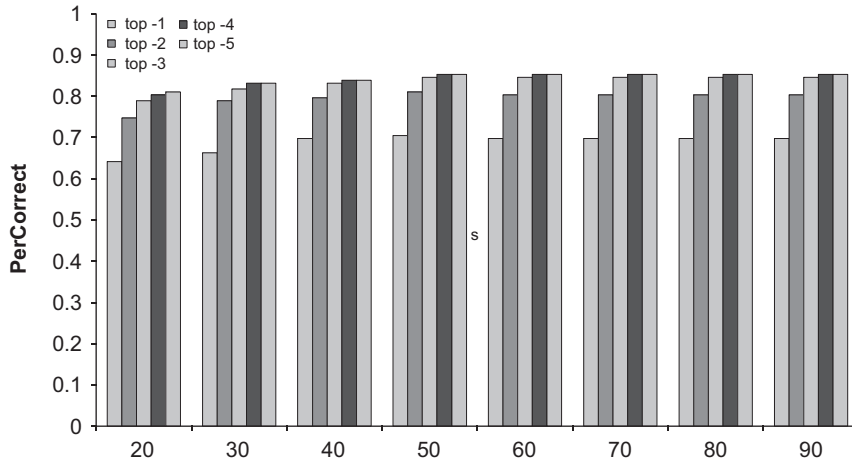


Fig. 9. Impact of *maxPage* on PerCorrect for TREC queries.

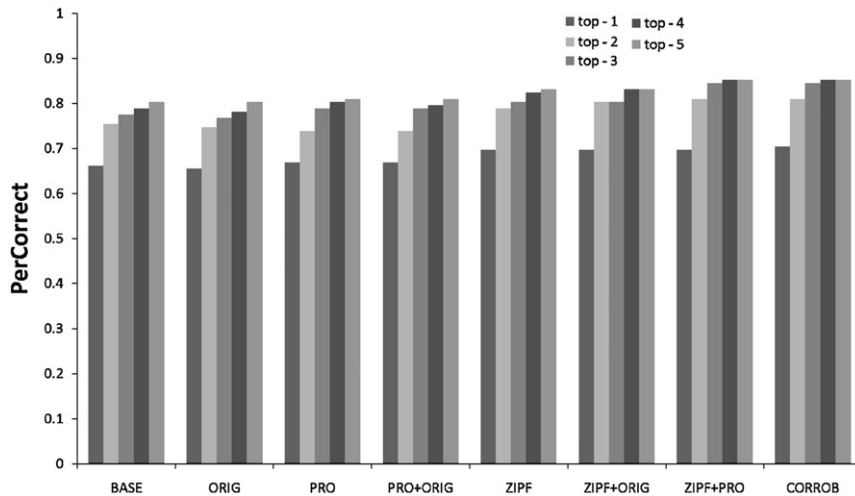


Fig. 10. Impact of scoring components of TREC queries.

$\alpha: s(p) = (1-\alpha)^{r(p)-1}$, where $s(p)$ is the score of page p and $r(p)$ is the rank of the page as returned from the search engine. Our new CORROB strategy outperforms ALPHA, with a p -value of 0.011. By using Zipf's law to model the decrease in page relevance, the CORROB method is able to return higher quality answers than ALPHA.

5.2.4. Number of pages retrieved

We dynamically retrieve pages as we corroborate answers (Section 4). Fig. 12 shows the average number of pages retrieved when answering queries extracted from TREC for the ALPHA method and the CORROB technique with the s parameter ranging from 0.6 to 2.0. Increasing the value of s in our CORROB method gives more weight to pages ranked higher in the search engine query result therefore reducing the number of pages needed to identify the top corroborated answers. In contrast, the ALPHA method has the highest average number of page retrieved except for $s=0.6$.

Table 4

Answer quality of scoring components.

	Combined	Numerical	Factoid
BASE	0.722	0.602	0.773
ORIG	0.717	0.590	0.770
PRO	0.727	0.610	0.776
ORIG+PRO	0.726	0.612	0.775
ZIPF	0.757	0.624	0.812
ZIPF+ORIG	0.760	0.625	0.817
ZIPF+PRO	0.766	0.657	0.812
CORROB	0.772	0.657	0.820

5.2.5. Time cost

The query evaluation time for the ALPHA and CORROB corroboration techniques is shown in Fig. 13. In particular, we divide the time cost into three parts: web page retrieval, answer extraction and answer corroboration.

The first part is the time cost for retrieving cached web pages from the search engine server, and the second and third parts are the time cost for answer extraction and corroboration, respectively. As expected, based on the number of pages retrieved (Fig. 12), the CORROB method outperforms ALPHA method for all values of s but $s = 0.6$. The CORROB method takes a bit more time than ALPHA with an s value of 0.6 due to more pages being retrieved.

While the overall query answering cost can be high, most of the time is spent on retrieving pages from the web. Our query evaluation time is reasonable (3 s for answer extraction and 0.4 s for corroboration on average for each query), and user may find it acceptable to wait for corroborated answers if it saves them the hassle of manually checking the pages themselves. In addition, our implementation does not have direct access to the search engine indexes and cache but must access these through an interface which incurs web retrieval costs. If our techniques were implemented within a search engine, they would provide much faster query response time.

5.3. MSN queries

We now report on experimental results over the queries extracted from the MSN Live Search query log. We set the default values of s , β , $maxPage$ to 1, 0.5, 50, respectively, as they provided the best quality results for a reasonable time cost over TREC queries.

5.3.1. Comparison with user click

We do not have a list of correct answers to the queries we extracted from the MSN Live Search query log. To evaluate the quality of our answers, we compare user clicks with pages that we used to generate the top corroborated answers. The internet being dynamic, by the time we performed our experiments, much of the information in the log was obsolete. Many pages on which users clicked were not available in the search engine (SE) query result anymore. Among the 331 pages

that appeared in user clicks, only 93 pages were still in the search engine result when we run the experiment. In addition, the position of these pages in the search engine result has changed greatly: while they had high positions when the query was issued (with average rank reverse of 0.5), they were ranked much lower in the current search engine result (with average rank reverse of 0.081). Overall, out of 138 queries in the MSN query set, for 81 of them the user clicks are no longer in the current search engine result.

Despite these limitations, we found that our corroborated answers correlate with user clicks. Fig. 14 shows, for each of the 57 queries that have at least one user click in the current search engine result, the number of user clicks, the number of corresponding pages that were returned in the search engine query result at the time of our experiments, the number of such pages that contained answers to the query, and the number of pages that contained the top corroborated answer. As shown, when a page on which the user clicked appears in the search engine query result, it contains one of the top-5 corroborated answers for 61% of the queries, and the top-1 answer for 42%. The results show that the corroboration approach is good at identifying answers that are of interest to the user, as many user clicks correlate with top corroborated answers. We looked into the remaining 22 queries for which there is no overlap between the user clicks and the pages we use to generate

Table 5

MRR comparison with existing question answering techniques.

	MRR
P-FREQ	0.663
A-FREQ	0.664
TOP-PAGE	0.360
BASE	0.722
ALPHA ($\alpha = 0.05$)	0.722
CORROB	0.772

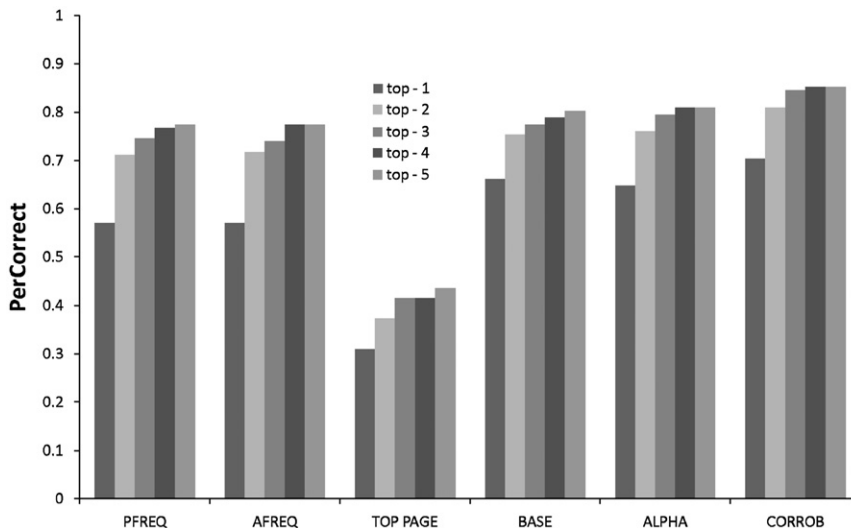


Fig. 11. PerCorrect comparison with existing question answering techniques.

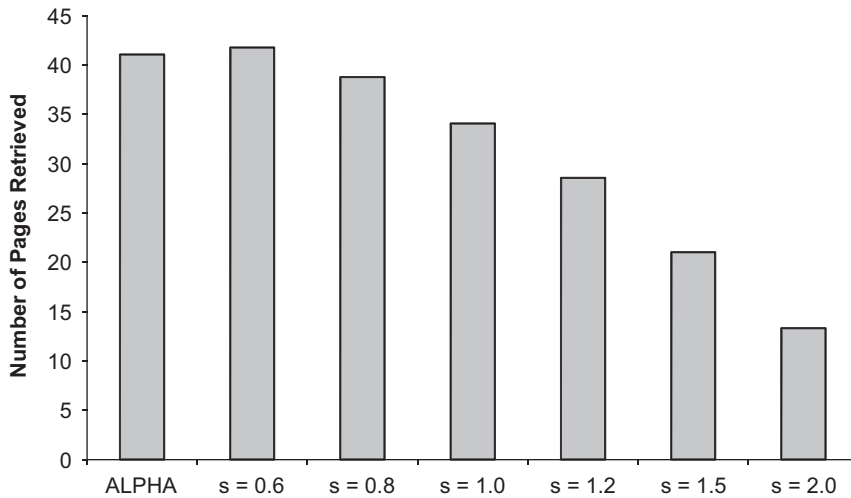


Fig. 12. Average number of pages retrieved for TREC queries.

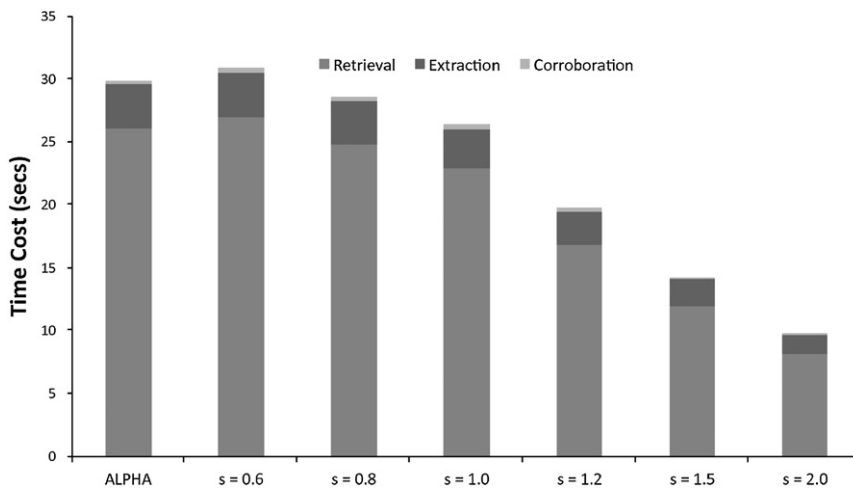


Fig. 13. Time cost of different scoring strategies for TREC queries.

top corroborated answers. We found that for 14 of them we could not extract answers from these user clicks, for five of them our corroboration terminated before reaching these user clicks because they were ranked much lower by the current search engine. For only three of these queries, our corroborated answers do not include answers extracted from user clicks. We believe that with a more recent search log, our corroboration method would show better similarity with the user clicks.

5.3.2. Number of pages retrieved

We also tested the number of pages retrieved for each MSN query. Our corroboration techniques dynamically retrieve web pages. On average, we retrieve 36.1 pages for each MSN query, which is lower than our 50 *maxPage* limit. The retrieval stopped dynamically before the *maxPage* value for 88% of the queries tested; in some cases fewer than 10 pages were enough to identify the best corroborated answer.

5.3.3. Time cost

Figs. 15–17 show the time cost for MSN queries that have at least one click in the current SE result and that have no clicks in the current SE result, respectively. (In order to display properly, we break the figure for the queries that have no clicks in the current SE result into two subfigures.) As with TREC queries, the retrieving time is the dominant factor in the total time cost. We also compare this time cost with the time the user spent on browsing the web pages. The MSN query log comes with an accurate timestamp for both the time when the query was issued by the user and the time when the user clicked on the web page. We calculate the user time cost as the duration between the time the query was issued and the time of the last user click. Of course, this is an approximation of the time the user took to find an answer to the query as it does not take into account the time the user spent loading and reading the corresponding pages nor does it guarantee that the user found an answer

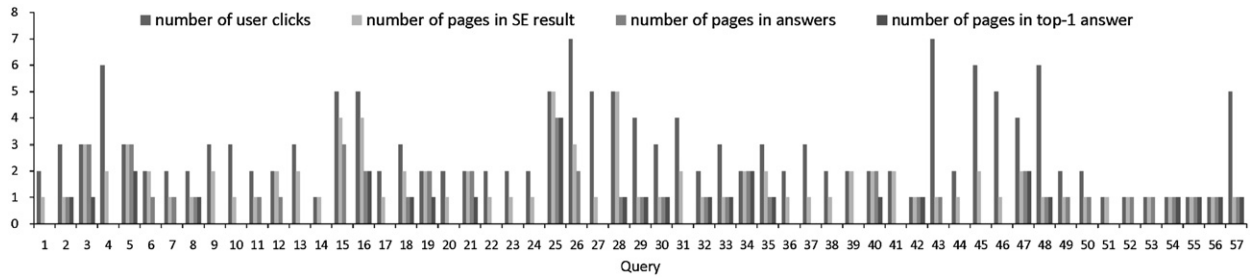


Fig. 14. Comparison between user clicks and corroborative answers for MSN queries.

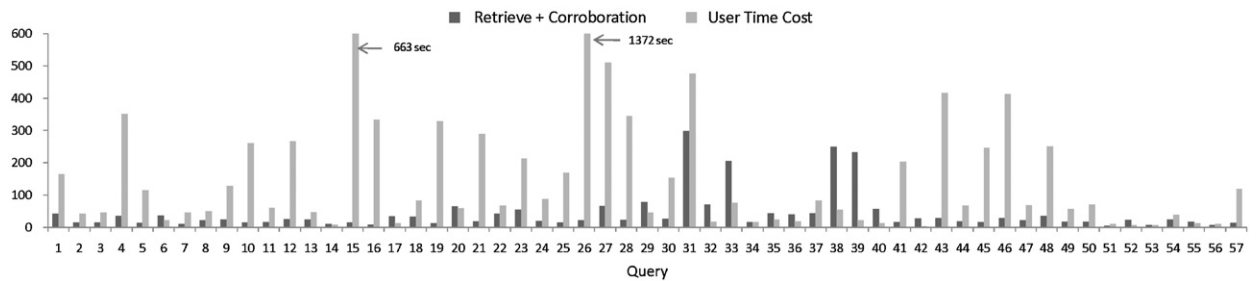


Fig. 15. Time cost for MSN queries that have at least one click in the current SE result.

within the clicked pages. On average, the user time cost (107.3s) is about 4.1 times more than the cost of our corroboration approach (26.4s). This indicates that our corroboration method is efficient in identifying the answers and can save users a great amount of time. Figs. 15–17 show that we do have a few queries for which our corroboration techniques take longer to return answers than the user time cost (e.g., Query 38, 39 in Fig. 15; Query 21, 35 in Fig. 16 and Query 41, 70, 76 in Fig. 17). For these queries our corroboration needs to retrieve more pages while the corresponding number of user clicks is relatively small. On average, the number of pages retrieved for MSN queries set is 36.1 and the number of user clicks for each query is 2.4. For the queries that our corroboration techniques take longer than the user time cost, the average number of pages retrieved is 38.3 and the average number of user clicks is 1.7. Typically, the queries that require expensive corroboration times are the ones for which multiple correct answers exist. For instance, for the query “who invented the television”, both “Farnsworth” and “Zworykin” are correct answers. The existence of multiple correct answers leads our corroboration techniques to go deeper into the search engine results to find the top-1 answer, therefore resulting in more pages being retrieved.

6. Related work

Several works have addressed the issue of extracting answers from web pages [25,2,29,8] for question answering. However, most of these question answering systems consider extracted answers separately and do not perform any answer corroboration. An exception is the Mulder

system [29], which uses frequency of answers to increase answer scores. Their approach is similar to our page-frequency approach of Section 5.2.3, which is outperformed by a corroborative strategy in our model. [36] presents a QA system which learns how to detect and rank answer passages by recovering a structured relational query from the posed question. However, their approach assumes that a large volume of training QA pairs are available.

Other works have considered the frequency of an answer as a measure of answer quality [13,14,5]. These models consider corroboration at the extraction-rule level, i.e., the score of an answer is increased if it is extracted from several high-quality rules or query rewrites, but not at the source level. Downey et al. [13] propose a probabilistic model to estimate the impact of repeated extraction from the web. The AskMSR system [14,5] uses several query rewrites and n-gram extraction, to improve the quality of its extracted answers. Dumais et al. propose a question answering system that uses web search engines as a backend retrieval system. This work is the closest to ours, as it considers answer redundancy in its scoring strategy. However, the redundancy is based on the number of answers returned by different query rewrites, and does not consider the quality of sources reporting the answer, or the quality of answers within the sources. In Section 5.2.3, we compared our corroboration techniques with an adaptation of an answer redundancy based approach (A-FREQ) to our model and show that corroboration significantly improves answer quality. Xu et al. [46] propose a novel approach containing a set of features of answer context to estimate the answer confidence extracted from documents. However, when applied to the Web corpora, their approach simplifies to a

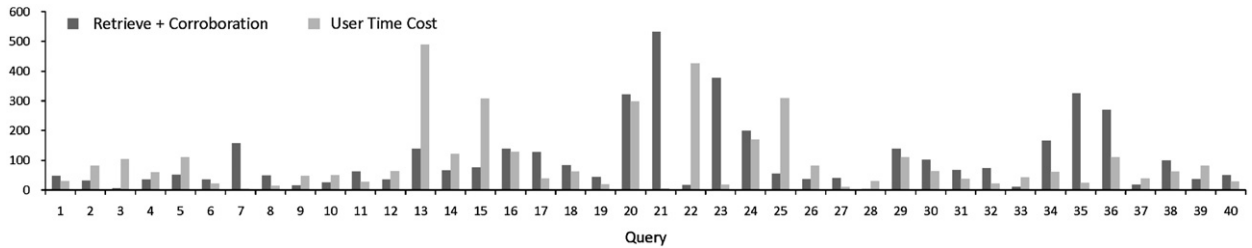


Fig. 16. Time cost for MSN queries that have no clicks in the current SE result (Part 1).

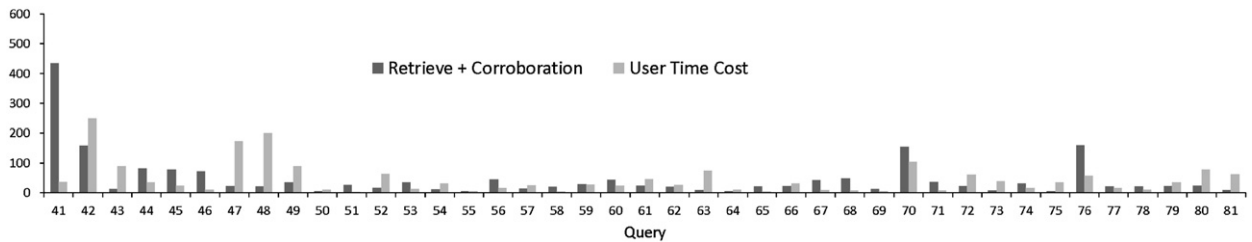


Fig. 17. Time cost for MSN queries that have no clicks in the current SE result (Part 2).

frequency-based approach. Zhang [49] proposes a novel approach that assigns scores to answers by comparing the query and the snippets from which the answers are extracted. In particular, for each answer, the snippets from which the answer is extracted are clustered and a bag-of-words feature vector is constructed for the answer. The answer score is then computed using the feature vector of the cluster and the query. However, his approach considers all the source snippets equally helpful and could be ineffective when a large number of low-quality sources are present.

Recent work has focused on identifying entities from large data collections and answering queries on these entities. In particular, the WISDM system [12] focuses on entity search and proposes a probabilistic framework to rank entities extracted from web pages by considering the context around entities. The NAGA system [24] provides semantic knowledge to improve web search engine results. Neither systems operate on live web data, but on indexes built over web-extracted data. Our corroboration system bears some similarity with the problem of *entity finding*, which has been studied since the introduction of TREC entity track in 2009. The task of entity finding is, given a query containing an input entity with its name and homepage, to find related entities that are of a target type. Fang et al. [17] proposed a hierarchical model for entity finding by considering the similarity between the query keyword and the document as well as the passage, from which entity candidates are extracted. Since they retrieve documents based on the query keywords, it is likely that a certain portion of the returned documents are duplicated pages. However, their approach does not consider source originality, which may result in assigning high scores to entities extracted from the duplicated pages. Moreover, our corroboration system focuses on a broader range of queries (e.g., Table 1), which is not limited to entities.

Yin et al. [48] propose the *Veracity* problem which studies how to find true facts from a large amount of conflicting information. By utilizing the relationships between web sites and their information, their algorithm, called *TRUTHFINDER*, can successfully find true facts among conflicting information. However, instead of using the web as the information provider, *TRUTHFINDER* relies on a small set of web sites from which to extract information and can only be used a specific information domain.

To the best of our knowledge, our techniques are the first to combine various measures of the trustworthiness of web sources, as well as the quality of the answers within the sources, and the frequency of these answers within the search engine query result. Using corroboration as a measure of answer quality has recently been suggested in non-web scenarios [27] where corroborating information is used to identify good answers across multiple databases in the presence of low-quality data.

Existing work on document redundancy has been explored by Bernstein and Zobel [6]. They explored syntactic techniques for identifying *content-equivalent* document pairs. In particular, they detect document redundancy by using document fingerprinting techniques, which has a proven record of application to large document collections.

Similar work on developing a unified framework that uses multiple resources for answer reranking has been proposed by Ko et al. [26]. In their framework, they boost answer scores by considering answer validation and answer similarity. However, ignoring the importance of the page from which the answer was extracted may expose their approach to malignant spammers, since pages are not equally trustworthy.

Metasearch has been studied extensively in literature to improve document retrieval results (for example,

[41,42,18,32,4]). Metasearch is the method of combining the output of multiple retrieving system by taking into consideration of the rank and relevance score of document in each retrieval system. Several models have been proposed for combining the outputs of multiple retrieval system. Thompson et al. propose the combination of expert opinion (CEO) model using the Bayesian model. Fox et al. propose the CombSUM and CombMNZ algorithm that considers the max, min, median and sum of the normalized relevance score of multiple retrieval system. Montague and Aslam propose models using the Borda Count and the Condorcet voting algorithm by considering only the rank of documents from each retrieval system. Although these models prove to be effective compared with using single retrieval system, they rely on the existence of multiple retrieval system and the availability of the relevance score of documents, which is not the case in our setting. In addition, metasearch system output a list of relevant documents but do not try to extract and aggregate answers from these documents. Our data corroboration approach could be used in conjunction with a metasearch system to increase the number, and relevance, of data sources considered in the corroboration.

We use existing information extraction [31] techniques for extracting answers from web pages. Our information extraction implementation is basic, considering only plain text content of the page. We plan to improve it by using structural information within the page using techniques similar to those described in [2,10,15].

Finally, work on top- k query processing algorithms has focused on adaptively reducing the amount of processing done by query evaluation techniques by ignoring data that would not be useful to identify the best answers to a query [16,30]. We use similar ideas in Section 4 to limit the number of pages we are retrieving from the search engine. However, we should note that standard top- k query processing techniques cannot be applied directly to our scenario as they consider each answer individually, and therefore can always easily identify the maximum possible score of an answer. In contrast, when allowing for corroboration, the maximum score of an answer can always increase. Anh et al. [3] develop top- k algorithm for text search aiming to optimize the retrieval when information from several (possibly sorted) lists needs to be merged. An interesting future direction would be to combine this work with our techniques to produce corroborated rankings from multiple lists.

7. Conclusion

We presented an approach to corroborate information from the web to improve search results. Our techniques use information extraction methods to identify relevant answers from a search engine result. We assign scores to each answer based on the frequency of the answer in the web search engine result, the relevance and originality of the pages reporting the answer, as well as the prominence of the answer within the pages. Our experimental evaluation on queries extracted from the TREC Question

Answering Track shows that a corroboration-based approach yields good quality answers. In addition, by comparing our approach to user-click behavior on a sample of queries from a MSN query log, we show that our techniques result in faster answers as they prevent users from having to manually check several sources.

Acknowledgement

This work was partially supported by a Microsoft Live Labs Search Award.

References

- [1] S.P. Abney, M. Collins, A. Singhal, Answer extraction, in: Proceedings of the Sixth Applied Natural Language Processing Conference (ANLP'00), 2000.
- [2] E. Agichtein, L. Gravano, Snowball: extracting relations from large plain-text collections, in: Proceedings of the Fifth ACM International Conference on Digital Libraries (DL'00), 2000.
- [3] V.N. Anh, O. de Kretser, A. Moffat, Vector-space ranking with effective early termination, in: Proceedings of the 24th Annual International ACM SIGIR Conference (SIGIR'01), 2001.
- [4] J.A. Aslam, M.H. Montague, Models for metasearch, in: Proceedings of the 24th Annual International ACM SIGIR Conference (SIGIR'01), 2001, pp. 275–284.
- [5] D. Azari, E. Horvitz, S. Dumais, E. Brill, Web-based question answering: a decision-making perspective, in: Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI'03), 2003.
- [6] Y. Bernstein, J. Zobel, Redundent documents and search effectiveness, in: Proceedings of 14th ACM International Conference on Information and Knowledge Management (CIKM'05), 2005.
- [7] L. Breslau, P. Cao, L. Fan, G. Phillips, S. Shenker, Web caching and zipf-like distributions: evidence and implications, in: Proceedings of the 18th IEEE International Conference on Computer Communications (INFOCOM'99), 1999.
- [8] E. Brill, S. Dumais, M. Banko, An analysis of the AskMSR question-answering system, in: Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing EMNLP02, 2002.
- [9] S. Brin, J. Davis, H. Garcia-Molina, Copy detection mechanisms for digital documents, in: Proceedings of the 1995 ACM International Conference on Management of Data (SIGMOD'95), 1995.
- [10] M.J. Cafarella, C. Re, D. Suciu, O. Etzioni, Structured querying of web text data: a technical challenge, in: Proceedings of the Third Biennial Conference on Innovative Data Systems Research CIDR07, 2007.
- [11] J. Chen, A. Diekema, M.D. Taffet, N.J. McCracken, N.E. Ozgencil, O. Yilmazel, E.D. Liddy, Question answering: Cnlp at the trec-10 question answering track, in: Proceedings of the 10th Text REtrieval Conference (TREC 2001), 2001.
- [12] T. Cheng, X. Yan, K.C.-C. Chang, Entityrank: searching entities directly and holistically, in: Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB'07), 2007.
- [13] D. Downey, O. Etzioni, S. Soderland, A probabilistic model of redundancy in information extraction, in: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05), 2005.
- [14] S. Dumais, M. Banko, E. Brill, J. Lin, A. Ng, Web question answering: is more always better?, in: Proceedings of the 25th Annual International ACM SIGIR Conference (SIGIR'02), 2002.
- [15] O. Etzioni, M.J. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D.S. Weld, A. Yates, Web-scale information extraction in KnowItAll, in: Proceedings of the 13th International Conference on the World Wide Web (WWW'04), 2004.
- [16] R. Fagin, A. Lotem, M. Naor, Optimal aggregation algorithms for middleware, *Journal of Computer and System Sciences (JCSS)* 66 (2003) 1.
- [17] Y. Fang, L. Si, Z. Yu, Y. Xian, Y. Xu, Entity retrieval with hierarchical relevance model, in: Proceedings of the 18th Text REtrieval Conference (TREC 2009), 2009.
- [18] E.A. Fox, M.P. Koushik, J.A. Shaw, R. Modlin, D. Rao, Combining evidence from multiple searches, in: TREC, 1992, pp. 319–328.

- [19] L. Gravano, P.G. Ipeirotis, N. Koudas, D. Srivastava, Text joins for data cleansing and integration in an rdbms, in: ICDE, 2003, pp. 729–731.
- [20] S.M. Harabagiu, D.I. Moldovan, M. Pasca, R. Mihalcea, M. Surdeanu, R.C. Bunescu, R. Girju, V. Rus, P. Morarescu, Falcon: boosting knowledge for answer engines, in: Proceedings of the Ninth Text REtrieval Conference (TREC-9), 2000.
- [21] E.H. Hovy, L. Gerber, U. Hermjakob, M. Junk, C.-Y. Lin, Question answering in webclopedia, in: Proceedings of the Ninth Text REtrieval Conference (TREC-9), 2000.
- [22] E.H. Hovy, U. Hermjakob, C.-Y. Lin, The use of external knowledge of factoid qa, in: TREC, 2001.
- [23] V. Jijkoun, M. de Rijke, Retrieving answers from frequently asked questions pages on the web, in: Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM'05), 2005.
- [24] G. Kasneci, F. Suchanek, M. Ramanath, G. Weikum, How naga coils: searching with entities and relations, in: Proceedings of the 16th International Conference on the World Wide Web (WWW'07), 2007.
- [25] B. Katz, Annotating the world wide web using natural language in: Proceedings of the Fifth RIAO Conference on Computer Assisted Information Searching on the Internet (RIAO'97), 1997.
- [26] J. Ko, L. Si, E. Nyberg, A probabilistic framework for answer selection in question answering, in: Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'07), 2007.
- [27] Y. Kotidis, A. Marian, D. Srivastava, Circumventing data quality problems using multiple join paths, in: Proceedings of the First International VLDB Workshop on Clean Database (CleanDB'06), 2006.
- [28] N. Koudas, A. Marathe, D. Srivastava, Flexible string matching against large databases in practice, in: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04), 2004, pp. 1078–1086.
- [29] C.C.T. Kwok, O. Etzioni, D.S. Weld, Scaling question answering to the web, in: Proceedings of the 10th International Conference on the World Wide Web (WWW'01), 2001.
- [30] A. Marian, N. Bruno, L. Gravano, Evaluating top-k queries over web-accessible databases, ACM Transactions on Database Systems (TODS) 29 (2004) 8.
- [31] A. McCallum, Information extraction: distilling structured data from unstructured text, ACM Queue 3 (9) (2005) 48–57.
- [32] M.H. Montague, J.A. Aslam, Condorcet fusion for improved retrieval, in: Proceedings of 11th ACM International Conference on Information and Knowledge Management (CIKM'02), 2002, pp. 538–548.
- [33] M. Pasca, S.M. Harabagiu, High performance question/answering, in: Proceedings of the 24th Annual International ACM SIGIR Conference (SIGIR'01), 2001.
- [34] J.M. Prager, E.W. Brown, A. Coden, D.R. Radev, Question-answering by predictive annotation, in: Proceedings of the 23rd Annual International ACM SIGIR Conference (SIGIR'00), 2000.
- [35] D.R. Radev, W. Fan, H. Qi, H. Wu, A. Grewal, Probabilistic question answering on the web, in: Proceedings of the 11th International Conference on World Wide Web (WWW'02), 2002, pp. 408–419.
- [36] G. Ramakrishnan, S. Chakrabarti, D. Paranjpe, P. Bhattacharyya, Is question answering an acquired skill?, in: Proceedings of 13th International Conference on World Wide Web (WWW'04), 2004.
- [37] S. Somasundaran, T. Wilson, J. Wiebe, V. Stoyanov, Qa with attitude: exploiting opinion type analysis for improving question answering in on-line discussions and the news, in: Proceedings of the International Conference on Weblogs and Social Media, 2007.
- [38] S. Tata, J.M. Patel, Estimating the selectivity of *tf-idf* based cosine similarity predicates, SIGMOD Record 36 (2) (2007) 7–12.
- [39] M. Theobald, J. Siddharth, A. Paepcke, Spotsigs: robust and efficient near duplicate detection in large web collections, in: Proceedings of the 31st Annual International ACM SIGIR Conference (SIGIR'08), 2008, pp. 563–570.
- [40] M. Theobald, G. Weikum, R. Schenkel, Top-k query evaluation with probabilistic guarantees, in: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04), 2004.
- [41] P. Thompson, A combination of expert opinion approach to probabilistic information retrieval, part 1: the conceptual model, Information Processing Management 26 (3) (1990) 371–382.
- [42] P. Thompson, A combination of expert opinion approach to probabilistic information retrieval, part 2: mathematical treatment of ceo model 3, Information Processing Management 26 (3) (1990) 383–394.
- [43] TREC. TREC Question Answering Track.
- [44] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 1 (6) (1945) 80–83.
- [45] M. Wu, A. Marian, Corroborating answers from multiple web sources, in: Proceedings of 10th International Workshop on Web and Database (WebDB'07), 2007.
- [46] J. Xu, A. Licuanan, J. May, S. Miller, R. Weischedel, Trec2002 qa at bbn: answer selection and confidence estimation, in: Proceedings of the 11th Text REtrieval Conference (TREC02), 2002.
- [47] H. Yang, T.-S. Chua, Qualifier: question answering by lexical fabric and external resources, in: Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics, 2003, pp. 363–370.
- [48] X. Yin, J. Han, P.S. Yu, Truth discovery with multiple conflicting information providers on the web, in: Proceedings of the 13th ACM International Conference on Knowledge Discovery and Data Mining (KDD'07), 2007.
- [49] D. Zhang, Web based question answering with aggregation strategy, in: Proceedings of the Sixth Asia-Pacific Web Conference, 2004, pp. 353–362.
- [50] G. Zipf, Selective Studies and the Principle of Relative Frequency in Language, Harvard University Press, 1932.