Web-Based List Question Answering

Hui Yang, Tat-Seng Chua
School of Computing
National University of Singapore
3 Science Drive 2, 117543, Singapore
yangh@lycos.co.uk, chuats@comp.nus.edu.sg

Abstract

While research on question answering has become popular in recent years, the problem of efficiently locating a complete set of distinct answers to list questions in huge corpora or the Web is still far from being solved. This paper exploits the wealth of freely available text and link structures on the Web to seek complete answers to list questions. We introduce our system, FADA, which relies on question parsing, web page classification/clustering, and content extraction to find reliable distinct answers with high recall.

1 Introduction

The Text REtrieval Conference Series (TREC) has greatly encouraged Question Answering (QA) research in the recent years. The QA main task in the recent TREC-12 involved retrieving short concise answers to factoid and list questions, and answer nuggets for definition questions (Voorhees, 2003). The list task in TREC-12 required systems to assemble a set of distinct and complete exact answers as responses to questions like *"What are the brand names of Belgian chocolates?"*. Unlike the questions in previous TREC conferences, TREC-12 list questions did not specify a target number of instances to return but expected all answers contained in the corpus. Current QA systems (Harabagiu et al., 2003; Katz et al., 2003) usually extract a ranked list of factoid answers from the top returned documents by retrieval engines. This is actually the traditional way to find factoid answers. The only difference between answering list questions and factoid questions here is that list QA systems allow for *multiple answers*, whose scores are above a cut-off threshold.

An analysis of the results of TREC-12 list QA systems (Voorhees, 2003) reveals that many of them severely suffer from two general problems: *low recall and non-distinctive answers*. The median average F_1 performance of list runs was only 21.3% while the best performer could only achieve 39.6% (Table 1). This unsatisfactory performance exposes the limitation of using only traditional Information Retrieval and Natural Language Processing techniques to find an exhaustive set of factoid answers as compared to only one.

TREC-12 Run Tag	Avg F_1
LCCmainS03	0.396
nusmml03r2	0.319
MITCSAIL03c	0.134
isi03a	0.118
BBN2003B	0.097
Average	0.213

Table 1: TREC-12 Top 5 Performers (Voorhees, 2003)

In contrast to the traditional techniques, the Web is used extensively in systems to rally round factoid questions. QA researchers have explored a variety of uses of the Web, ranging from surface pattern mining (Ravichandran et al., 2002), query formulation (Yang et al., 2003), answer validation (Magnini et al., 2002), to directly finding answers on the Web by data redundancy analysis (Brill et al., 2001). These systems demonstrated that with the help of the Web they could generally boost baseline performance by 25%-30% (Lin 2002).

The well-known redundancy-based approach identifies the factoid answer as an N-gram appearing most frequently on the Web (Brill et al. 2001). This idea works well on factoid questions because factoid questions require only one instance and web documents contains a large number of repeated information about possible answers. However, when dealing with list questions, we need to find all distinct instances and hence we cannot ignore the less frequent answer candidates. The redundancy-based approach fails to spot novel or unexpectedly valuable information in lower ranked web pages with few occurrences.

In this paper, we propose a novel framework to employ the Web to support list question answering. Based on the observations that multiple answer instances often appear in the list or table of a single web page while multiple web pages may also contain information about the same instance, we differentiate these two types of web pages. For the first category, which we call Collection Page (CP), we need to extract table/list content from the web page. For the second category, which we call Topic Page (TP), we need to find distinct web pages relating to different answer instances. We will demonstrate that the resulting system, FADA (**F**ind **A**ll **D**istinct **A**nswers), could achieve effective list question answering in the TREC corpus.

The remainder of this paper is organized as following. Section 2 gives the design considerations of our approach. Section 3 details our question analysis and web query formulation. Section 4 describes the web page classification and web document features used in FADA. Section 5 shows the algorithm of topic page clustering while Section 6 details the answer extraction process. Section 7 discusses experimental results. Section 8 concludes the paper.

2 Design Considerations

Our goal is to find as many distinct exact answers on the Web as possible. This requires us to:

- perform effective and exhaustive search; and
- extract distinct answers.

In order to perform effective search, we employ question transformation to get effectual web queries. However, this is not a trivial task. If the query is too general, too many documents may be retrieved and the system would not have sufficient resources to scan through all of them. If the query is too specific, no pages may be retrieved.

Given millions of web pages returned by search engines, our strategy is to divide-and-conquer by first identify Collection Pages (CP) that contain a list of answer instances. For example, for the question *"What breeds of dog have won the "Best in Show" award at the Westminster Dog Show?"*, we can find a Collection Page as shown in Figure 1 (top). Such a web page is a very good resource of answers. In general, we observe that there is a large number of named entities of the type desired appearing in a Collection Page, typically in a list or table. Our intuition is that if we can find a Collection Page that contains almost all the answers, then the rest of the work is simply to extract answers from it or related web pages by wrapper rule induction.

Another kind of "good" web page is a Topic Page, that contains just one answer instance (Figure 1, bottom). It typically contains many named entities, which correspond to our original query terms and some other named entities of the answer target type. Given the huge amount of web data, there will be many Topic Pages that refer to the same answer instance. There is hence a need to group the pages and to identify a pertinent and distinctive page in order to represent a distinct answer.

Web page class	Description
Collection Page	Containing a list of answers
Topic Page	The best page to represent an answer instance
Relevant Page	Relevant to an answer instance by providing either support or objection to the Topic Page
Irrelevant Page	Not related to any answer instance

Table 2: Web Page Classes

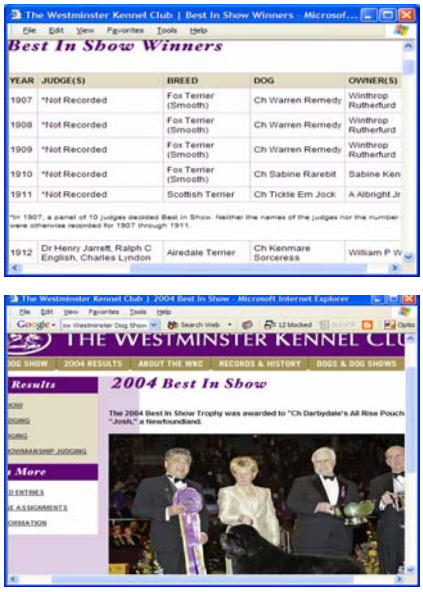

Figure 1: Examples of Collection Page (top) and Topic Page (bottom)

The rest of the top returned web pages could be either relevant or irrelevant to the question. In summary, we need to classify web pages into four classes: *Collection Page, Topic Page, Relevant Page, and Irrelevant Page* (Table 2), based on their functionality and contribution in finding list answers.

Based on the above considerations, we propose a general framework to find list answers on the Web using the following steps:

a) Retrieve a good set of web documents.
b) Identify Collection Pages and distinct Topic Pages as main resources of answers.
c) Perform clustering on other web pages based on their similarities to distinct Topic Pages to form clusters that correspond to distinct answer instances.
d) Extract answers from Collection Pages and Topic Page clusters.

3 Question Transformation and Web Page Retrieval

Agichtein et al. (2001) presented a technique on learning search engine specific query transformations for question answering. A set of transformation rules are learned from a training corpus and applied to the questions at the search time. Related work could also be found in Kwok et al. (2001) where the user's question is processed by a parser to learn its syntactic structure and various query modulation techniques are applied to the initial questions to get high quality results for later answer extraction.

FADA performs question parsing to identify key question words and the expected answer type. It extracts several sets of words from the original question and identifies the detailed question classes. It

then formulates a number of queries by combining the known facets together with heuristic patterns for list questions.

We perform both shallow and full parsing on a question followed by Named Entity Recognition (NER) to get the known query facets and their types. The shallow parser we used is the free online memory-based chunker[1] and the full parser is MINIPAR[2]. Both parsers are very efficient and usually parse 300 words within a second. The procedure of query parsing is as follows:

a) Remove head words. The head words in a question could be wh-question words and leading verbs. The list of head words includes "*who, what, when, where, which, how, how much, how many, list, name, give, provide, tell"*, etc. Removing them enables us to get the correct subject/object relation and verb in the question. For example, for question *"What breeds of dog have won the 'Best in Show' award at the Westminster Dog Show?"*, after removing the head word, the question becomes *"breeds of dog have won the 'Best in Show' award at the Westminster Dog Show"*.

b) Detect subject and object for the remaining question segments by shallow parsing. For example, after parsing the above question, we get:

```
[NP1 Subject breeds//NNS NP1 Subject] {PNP [P
of/IN P] [NP dog/NN NP] PNP} [VP1
have/VBP won/VBN VP1] [NP1 Object the/DT
``/`` Best/JJS NP1 Object] {PNP [P in/IN
P] [NP Show/NNP ''/'' award/NN NP]
PNP} {PNP [P at/IN P] [NP the/DT
Westminster/NNP Dog//NNP Show/NNP NP]
PNP}
```

From the parsed sentence, we want to get the logical subject as the sentence subject or its immediate modifiers. Here we have the logical subject-*"breeds of dog"*, verb-"*won*", and logical object-"*the best in show award*". If the resulting logical subject/object is the term "*that*" as in the following parsed query for "*U.S. entertainers that later became politicians*":

```
[NP U.S./NNP entertainers//NNS NP]
[NP1 Subject that/WDT NP1 Subject] [ADVP
later/RB ADVP] [VP1 became/VBD VP1]
[NP1 Object politicians//NNS NP1 Object]
```

we get the noun or noun phrase before the clause as the logical subject/object. Hence, we have the logical subject-*"entertainers"*, action-*"became"*, and logical object-"*politician*".

c) Extract all the noun phrases as potential descriptions from the remaining question segments, which are usually prepositional phrases or clauses. For the "dog breeds" example, we get the descriptions-*"Westminster Dog Show"*.

d) Apply named entity recognition to the resulting description phrases by using NEParser, a fine-grained named entity recognizer used in our TREC-12 system (Yang et al., 2003). It assigns tags like "person", "location", "time", "date", "number". For the "dog breed" example, "*Westminster*" gets the location tag.

After the above analysis, we obtain all the known facets provided in the original question. We then make use of this knowledge to form web queries to get the right set of pages. This is a crucial task in dealing with the Web. One of the query transformation rules is given as follows:

```
(list|directoty|category|top|favorite
)? (:|of)? <subj> <action>? <object>?
<description1>? <description2>? …
<descriptionN>?
```

The rule starts the query optionally with leading words (list, directory, category), optionally followed by a colon or "of", followed by subject phrase (<subj>), optionally followed by action (<action>), optionally followed by object (<object>) and description phrases (<description1>…<descriptionN>). In the above pattern, "?" denotes optional, "…" omit, and "|" alternative. For example, for the "*dog breed*" question, we form queries *"breed of dog won best in show Westminster Dog Show"*, *"directory breed of dog best in show Westminster Dog Show"*, and *"list breed of dog won best in show" etc.*

Transforming the initial natural language questions into a good query can dramatically improve the chances of finding good answers. FADA submits these queries to well-known search engines (Google, AltaVista, Yahoo) to get the top *1,000* Web pages per search engine per query. Here we attempt to retrieve a large number of web pages to serve our goal - find *All Distinct* answers. Usually, there are a large number of web pages which are redundant as they come from the same URL addresses. We remove the redundant web pages using the URL addresses as the guide. We also filter out files whose formats are neither HTML nor plain text and those whose lengths are too short or too long. Hence the size of the resulting document set for each question varies from a few thousands to ten of thousands.

4 Web Page Classification

In order to group the web pages returned by the search engines into the four categories discussed earlier, it is crucial to find a good set of features to represent the web pages. Many techniques such as td.idf (Salton and Buckley, 1988) and a stop word list have been proposed to extract lexical features to help document clustering. However, they do not work well for question answering. As pointed out by Ye et al. (2003) in their discussion on the person/organization finding task, given two resume

[1] http://ilk.kub.nl/cgi-bin/tstchunk/demo.pl
[2] http://www.cs.ualberta.ca/~lindek/minipar.htm

pages about different persons, it is highly possible that they are grouped into one cluster because they share many similar words and phrases. On the other hand, it is difficult to group together a news page and a resume page about the same target entity, due to the diversity in subject matter, word choice, literary styles and document format. To overcome this problem, they used mostly named entity and link information as the basis for clustering. Compared to their task, our task of finding good web documents containing answers is much more complex. The features are more heterogeneous, and it is more difficult to choose those that reflect the essential characteristics of list answers.

In our approach, we obtain the query words through subject/object detection and named entity recognition. We found that there are a large number of named entities of the same type appearing in a Collection Page, typically within a list or table. And in a Topic Page, there is also typically a group of named entities, which could correspond to our original query terms or answer target type. Therefore, named entities play important roles in semantic expression and should be used to reflect the content of the pages.

The Web track in past TREC conferences shows that URL, HTML structure, anchor text, hyperlinks, and document length tend to contain important heuristic clues for web clustering and information retrieval (Craswell and Hawking, 2002). We have found that a Topic Page is highly likely to repeat the subject in its URL, title, or at the beginning of its page. In general, if the subject appears in important locations, such as in HTML tags <title>, <H1> and <H2>, or appears frequently, then the corresponding pages should be Topic Pages and their topic is about the answer target.

Followed the above discussion, we design a set of 29 features based on Known Named Entity Type, Answer Named Entity Type, ordinary Named Entities, list, table, URL, HTML structure, Anchor, Hyperlinks, and document length to rep resent the web pages. Table 3 lists the features used in our system. In the table and subsequent sections, NE refers to Named Entity.

We trained two classifiers: the Collection Page classifier and the Topic Page classifier. The former classifies web pages into Collection Pages and non-collection pages while the later further classifies the non-collection pages into Topic Pages and Others. Both Classifiers are implemented using Decision Tree C4.5 (Quinlan 1993). We used 50 list questions from TREC-10 and TREC-11 for training and TREC-12 list questions for testing. We parse the questions, formulate web queries and collect web pages by using the algorithm described in Section 2.

	Feature	Meaning
1	|PER|	# of Person NEs
2	|ORG|	# of Organization NEs
3	|LOC|	# of Location NEs
4	|TME|	# of Time NEs, including date, year, month
5	|NUM|	# of Numer NEs
6	|COD|	# of Code NEs, including phone number, zip code, etc
7	|OBJ|	# of Object NEs, including animal, planet, book, etc
8	|NE|	Total number of the above NEs
9	|Known_NE|	Total # of NEs within the same NE type as in the question. In the "dog breed" example, it is the number of Location NEs since "Westminster" is identified as Location by NER.
10	|Unknown_NE|	# of NEs belonging to other NE type. In the "dog breed" example, it is the total number of *Time and Breed* NEs
11	|Answer_NE|	# of NEs belonging to expected answer type. In the "dog breed" example, it is the number of *Breed* NEs
12	|Known_NE| / |NE|	Ratio of | Known _NE| to |NE|
13	|Unknown_NE| / |NE|	Ratio of | Unknown _NE| to |NE|
14	|Answer_NE| / |NE|	Ratio of |Answer_NE| to |NE|
15	Length	# of tokens in a page
16	Content_Length	# of words in a page excluding HTML tags
17	|NE|/Length	Ratio of |NE| to |Token|
18	|NE|/Content_Length	Ratio of |NE| to |Word|
19	|In_Link|	# of in-links
20	|Out_Link|	# of out-links
21	|All_Link|	The sum of in-links and out-links
22	Keyword_in_Title	Boolean indicating presence of keywords in page title
23	Keyword_in_URL	Boolean indicating presence of keywords in URL
24	Keyword_in_Page	Boolean indicating presence of keywords in the page
25	|Answer_NE_in_Title|	# of NEs belonging to expected answer type presenting in page title
26	|Answer_NE_in_URL|	# of NEs belonging to expected answer type presenting in URL
27	| |	# of HTML tags representing a list or table, including , , , ,<td>
28	| <a href=|	# of HTML tags, including , , , ,<td> to represent a list/table of anchors,
29	URL_Depth	The depth of URL

Table 3: Web Page Features

Each sample is represented using the features listed in Table 3. Some of the decision rules are as follows:

```
a) OUT_Link >= 25 & NE > 78 &
b) Answer_NE >= 30 -> Class CP OUT_Link
   <= 25 & Answer_NE <= 5 & NE > 46 ->
   Class TP
c) OUT_Link >= 25 & URL_Depth > 3 ->
   Others
d) NE <= 4  -> Others
```

Rule a) implies that good Collection Pages should have many outlinks, NEs and especially answer NEs. Rule b) implies that good Topic Pages should have many NEs but relatively few links and answer NEs. Rule c) show that Others have deeper URL depth; while Rule d) shows that they have fewer NEs.

Web page classification enables us to get Collection Pages, Topic Pages and the rest of the pages. Our experiments on TREC-12 list questions showed that we can achieve a classification precision of 91.1% and 92% for Collection Pages and Topic Pages respectively.

5 Finding Answer Sources

Based on Web page classification, we form the initial sets of Collection Pages CPSet, Topic Pages TPSet and OtherSet. In order to boost the recall, we first use the outgoing links of Collection Pages to find more Topic Pages. These outgoing pages are potential Topic Pages but not necessarily appearing among the top returned web documents. Our subsequent tests reveal that the new Topic Pages introduced by links from Collection Pages greatly increase the overall answer recall by 23%. The new Topic Page set becomes:

```
TPSet' = TPSet + {outgoing pages of CPs}
```

Second, we select distinct Topic Pages. We compare the page similarity between each pair of Topic Pages using the algorithm below.

```
for each pair {tp_i, tp_j} in TPSet'
   if (sim(tp_i,tp_j)> Θ)
      if ℵ_ANE_in_tp_i > ℵ_ANE_in_tp_j
           move tp_j into OtherSet;
```

Here the page similarity function *sim()* is a linear combination of overlaps between *Known_NE, Answer_NE, URL similarity and link similarity*. Θ is preset at 0.75 and may be overridden by the user. $\aleph_{ANE}\_in\_tp_i$ is the number of named entities of answer type in Topic Page tp_i. For those pairs with high similarity, we keep the page that contains more named entities of answer type in *TPSet'* and move the other into *OtherSet*. The resulting Topic Pages in *TPSet'* are distinct and will be used as cluster seeds for the next step.

Third, we identify and dispatch Relevant Pages from *OtherSet* into appropriate clusters based on their similarities with the cluster seeds.

```
for each rp_i in OtherSet {
   k = argmax {sim(rp_i , tp_k) }
   if (sim(rp_i , tp_k ) >  τ )
     insert rp_i into cluster_k;
   else
     insert rp_i into IrrelevantSet; }
```

Here τ is preset at 0.55, and *sim()* is defined as above. Each cluster corresponds to a distinct answer instance. The Topic Page provides the main facts about that answer instance while Relevant Pages provide supporting materials for the unique answer instance. The average ratio of correct clustering is 54.1% in our experiments.

Through web page clustering, we avoid early answer redundancy, and have a higher chance to finding distinct answers on the noisy Web.

6 Answer Extraction

6.1 HTML Source Page Cleaning

Many HTML web pages contain common HTML mistakes, including missing or unmatched tags, end tags in the wrong order, missing quotes round attributes, missed / in end tags, and missing > closing tags, etc. We use HtmlTidy[3] to clean up the web pages before classification and clustering. FADA also uses an efficient technique to remove advertisements. We periodically update the list from Accs-Net[4], a site that specializes in creating such blacklists of advertisers. If a link address matches an entry in a blacklist, the HTML portion that contained the link is removed.

6.2 Answer Extraction from CP

Collection Pages are very good answer resources for list QA. However, to extract the "exact" answers from the resource page, we need to perform wrapper rule induction to extract the useful content. There is a large body of related work in content extraction, which enables us to process only extracted content rather than cluttered data coming directly from the web. Gupta et al. (2003) parsed HTML documents to a Document Object Model tree and to extract the main content of a web page by removing the link lists and empty tables. In contrast, our link list extractor finds all link lists, which are table cells or lists for which the ratio of the number of links to the number of non-linked words is greater than a specific ratio. We have written separate extractors for each answer target type. The answers obtained in Collection Pages are then "projected" onto the TREC AQUAINT corpus to get the TREC answers (Brill et al., 2001).

6.3 Answer Extraction from TP Cluster

Having web pages clustered for a certain question, especially when the clusters nicely match distinct answer, facilitates the task of extracting the possible answers based on the answer target type. We perform this by first analyzing the main Topic Pages in each cluster. In case we find multiple passages containing different answer candidates in the same Topic Page, we select the answer candidate from the passage that has the most variety of NE types since it is likely to be a comprehensive description about different facets of a question topic. The answer found in the Topic Page is then "projected" onto the QA corpus to get the TREC answers as with the Collection Page. In case no TREC answers can be found

[3] http://htmltrim.sourceforge.net/tidy.html
[4] http://www.accs-net.com/hosts/get_hosts.html

based on the Topic Page, we go to the next most relevant page in the same cluster to search for the answer. The process is repeated until either an answer from the cluster is found in the TREC corpus or when all Relevant Pages in the cluster have been exhausted.

For the question "*Which countries did the first lady Hillary Clinton visit?*", we extracted the *Locations* after performing Named Entity analysis on each cluster and get 38 country names as answers. The recall is much higher than the best performing system (Harabagiu et al., 2003) in TREC-12 which found 26 out of 44 answers.

7 Evaluation on TREC-12 Question Set

We used the 37 TREC-12 list questions to test the overall performance of our system and compare the answers we found in the TREC AQUAINT corpus (after *answer projection* (Brill et al. 2001)) with the answers provided by NIST.

7.1 Tests of Web Page Classification

In Section 3, the web pages are classified into three classes: Collection Pages, Topic Pages, and Others. Table 4 shows the system performance of the classification. We then perform a redistribution of classified pages, where the outgoing pages from CPs go to TP collection, and the Relevant Pages are grouped as supportive materials into clusters, which are based on distinct Topic Page. Nevertheless, the performance of web page classification will influence the later clustering and answer finding task. Table 4 shows that we could achieve an overall classification average precision of 0.897 and average recall of 0.851. This performance is adequate to support the subsequent steps of finding complete answers.

Page Class	Avg Prec.	Avg Rec.
Collection	91.1%	89.5%
Topic	92.0%	88.4%
Relevant	86.5%	83.4%
Overall	89.7%	85.1%

Table 4: Performance of Web Page Classification

7.2 Performance and Effects of Web Page Clustering

Relevant Pages are put into clusters to provide supportive material for a certain answer instance. The performance of Relevant Page dispatch/clustering is 54.1%. We also test different clustering thresholds for our web page clustering as defined in Section 5. We use the F1 measure of the TREC-12 list QA results as the basis to compare the performance of different clustering threshold combinations as shown in xx. We obtain the best performance of F_1 = 0.464 when τ=0.55 and Θ=0.75.

	$\Theta(0.55)$	$\Theta(0.65)$	**$\Theta(0.75)$**	$\Theta=0.85$
$\tau=0.25$	0.130	0.234	0. 324	0.236
$\tau=0.35$	0.136	0.244	0. 338	0.232
$\tau=0.45$	0.148	0.332	0. 428	0.146
$\tau=0.55$	0.166	0.408	**0. 464**	0.244
$\tau=0.65$	0.200	0.322	0. 432	0.236

Table 5: Clustering Threshold Effects

7.3 Overall Performance

Table 6 compares a baseline list question answering system with FADA. The baseline is based on a system which we used in participation in the TREC-12 QA task (Yang et al., 2003). It extends the traditional IR/NLP approach for factoid QA to perform list QA, as is done in most other TREC-12 systems. It achieves an average F_1 of 0.319, and is ranked 2nd in the list QA task.

We test two variants of FADA – one without introducing the outgoing pages from CPs as potential TPs (FADA1), and one with (FADA2). The two variants are used to evaluate the effects of CPs in the list QA task. The results of these two variants of FADA on the TREC-12 list task are presented in Table 6.

	Avg P	*Avg R*	*Avg F_1*
Baseline	0.568	0.264	0.319
FADA1 (w/o outgoing pages)	0.406	0.344	0.372
FADA2 (w/ outgoing pages)	0.516	0.422	0.464
TREC-12 best run	-	-	0.396

Table 6: Performance on TREC-12 Test Set

Without the benefit of the outgoing pages from CPs to find potential answers, FADA1 could boost the average recall by 30% and average F_1 by 16.6% as compared to the baseline. The great improvement in recall is rather encouraging because it is crucial for a list QA system to find a complete set of answers, which is how list QA differ from factoid QA.

By taking advantage of the outgoing pages from CPs, FADA2 further improves performance to an average recall of 0.422 and average F_1 of 0.464. It outperforms the best TREC-12 QA system (Voorhees, 2003) by 19.6% in average F_1 score.

From Table 6, we found that the outgoing pages from the Collection Pages (or resource pages) contribute much to answer finding task. It gives rise to an improvement in recall of 22.7% as compared to the variant of FADA1 that does not take advantage of outgoing pages. We think this is mainly due to the characteristics of the TREC-12 questions. Most questions ask for well-known things, and famous events, people, and organization. For this kind of questions, we can easily find a Collection Page that contains tabulated answers since there are web sites that host and maintain such information. For instance, "Westminster Dog Show" has an official

web site[5]. However, for those questions that lack Collection Pages, such as "*Which countries did the first lady Hillary Clinton visit?*", we still need to rely more on Topic Pages and Relevant Pages.

With the emphasis on answer completeness and uniqueness, FADA uses a large set of documents obtained from the Web to find answers. As compared to the baseline system, this results in a drop in average answer precision although both recall and F_1 are significantly improved. This is due to the fact that we seek most answers from the noisy Web directly, whereas in the baseline system, the Web is merely used to form new queries and the answers are found from the TREC AQUAINT corpus. We are still working to find a good balance between precision and recall.

The idea behind FADA system is simple: Since Web knowledge helps in answering factoid questions, why not list questions? Our approach in FADA demonstrates that this is possible. We believe that list QA should benefit even more than factoid QA from using Web knowledge.

8 Conclusion

We have presented the techniques used in FADA, a system which aims to find complete and distinct answers on the Web using question parsing, web page classification/clustering and content extraction. By using the novel approach, we can achieve a recall of 0.422 and F_1 of 0.464, which is significantly better than the top performing systems in the TREC-12 List QA task. The method has been found to be effective. Our future work includes discovering answers on non-text web information, such as images. Much text information is stored as images on the web, and hence, cannot be accessed by our approach, and some do contain valuable information.

References

E. Agichtein, S. Lawrence, and L. Gravano. 2001. "Learning search engine specific query transformations for question answering." In the Proceedings of the 10th ACM World Wide Web Conference (WWW 2001).

E. Brill, J. Lin, M. Banko, S. Dumais, and A. Ng. 2001. "Data-intensive question answering". In the Proceedings of the 10th Text REtrieval Conference (TREC 2001).

N. Craswell, D Hawking. 2002. "Overview of the TREC-2002 Web Track", In the Proceedings of the 11th Text REtrieval Conference. (TREC 2002).

S. Gupta, G. Kaiser, D. Neistadt, P. Grimm, 2003. "DOM-based Content Extraction of HTML Documents", In the Proceedings of the 12th ACM World Wide Web conference. (WWW 2003).

S. Harabagiu, D. Moldovan, C. Clark, M. Bowden, J. Williams, J. Bensley, 2003 "Answer Mining by Combining Extraction Techniques with Abductive Reasoning," In the notebook of the 12th Text REtrieval Conference (TREC 2003), 46-53.

B. Katz, J. Lin, D. Loreto, W. Hildebrandt, M. Bilotti, S. Felshin, A. Fernandes, G. Marton, F. Mora, 2003, "Integrating Web-Based and Corpus-Based Techniques for Question Answering", In the notebook of the 12th Text REtrieval Conference (TREC 2003), 472-480.

C. Kwok, O. Etzioni, and D. S. Weld, 2001, "Scaling Question Answering to the Web", In the Proceedings of the 10th ACM World Wide Web conference. (WWW 2001).

C. Y. Lin, "The Effectiveness of Dictionary and Web-Based Answer Reranking." In the Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002).

B. Magnini, M. Negri, R. Prevete and H. Tanev. 2002. "Is it the Right Answer? Exploiting Web Redundancy for Answer Validation". In the Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. (ACL 2002), 425-432.

J. R. Quinlan, 1993. C4.5: Programs for Machine Learning. Morgan-Kaufmann, San Francisco.

D. Ravichandran, and E. H. Hovy. 2002. "Learning Surface Text Patterns for a Question Answering System." In the Proceedings of the 40th ACL conference. (ACL 2002).

G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval", Information Processing and Management: an International Journal, v.24 n.5, 1988

E.M.Voorhees. 2003. "Overview of the TREC 2003 Question Answering Track." In the notebook of the 12th Text REtrieval Conference (TREC 2003), 14-27.

H. Yang, T. S. Chua, S Wang, C. K. Koh. 2003. "Structured Use of External Knowledge for Event-based Open Domain Question Answering", In the Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003).

H. Yang, H. Cui, M. Maslennikov, L. Qiu, M. Y. Kan, T. S. Chua. 2003. "QUALIFIER in the TREC12 QA Main Task", In the notebook of the 12th Text REtrieval Conference (TREC 2003).

S. Ye, T. S. Chua, J. R. Kei. 2003. "Querying and Clustering Web Pages about Persons and Organizations". Web Intelligence 2003, 344-350.

[5] http://www.westminsterkennelclub.org/