ABSTRACT

Mobile devices are being used as aids to access a variety of information resources by browsing the web. However, given their limited screen real estate and soft keyboards, general web browsing to access information is a tedious task. A system that (a) allows a user to specify their information need as a spoken language query and (b) returns the answer to user’s information need directly would be particularly appealing for these devices. In this paper, we address these two problems and present techniques that model question-answering as a query retrieval task. We show that we improve the retrieval accuracy by tightly integrating the constraints of the speech recognition and search components.

Index Terms— Voice search, Question-Answering, Finite-state Transducers, Tight coupling

1. INTRODUCTION

Mobile devices are increasingly being used as aids to access information and accomplish complex tasks, extending beyond their use as human-human communication devices. With the ability to browse the Internet using these devices, potentially unlimited amounts of information is accessible at our fingertips. This is in spite of the soft keyboards and the limited screen space that make these devices cumbersome to type in text input compared to conventional desktop and laptop computers.

Here, we address two problems. First, we address the problem of information access on a mobile device, beyond browsing of web pages to directly answer the user’s question. This is in contrast to the prevalent approach where a user types in a query using keywords to a search engine, browses the returned results on the small screen to select a potentially relevant document, suitably magnifies the screen to view the document and searches for the answer in the document. By providing a method for the user to pose her query in natural language and presenting the relevant answer(s) to her question, we expect the user’s information need to be fulfilled sooner.

The second problem that we address in this paper relates to the input modality. With small screens and soft/small keyboards, text input into mobile devices is cumbersome. In addition, by the mobile nature of these devices, users often would like to use them in hands-busy environments, ruling out the possibility of typing text. We address this issue by allowing the user to query the information repository using speech. We expect that spoken language queries to be a more natural and less cumbersome way of interacting with mobile devices.

We present a system for speech-driven question answering as a solution toward addressing these two issues. The system provides a natural input modality – spoken language input – for the users to pose their information need and presents a collection of answers that potentially address the information need directly. In contrast to the text-based question-answer retrieval task [1], where most of the systems attempt to synthesize the answers from unstructured text, we retrieve the related questions and their answers from a archive of human generated answers to questions. This ensures higher accuracy of the answers retrieved (if found in the archive) and also allows us to retrieve related questions on the user’s topic of interest.

The layout of the paper is as follows. In Section 2 we review the related literature. In Section 3 we illustrate the system for speech-driven question answering. We present the retrieval methods we used to implement the system in Section 4. We evaluate the retrieval methods using text and speech queries and present the results in Section 6. We conclude in Section 7.

2. RELATED WORK

There are several threads of related literature to the work presented in this paper. The work on text-based question-answering (comprehensively summarized in [1]) ranges widely in terms of linguistic sophistication. At one end of the spectrum, there are linguistically motivated systems [2, 3] that analyze the user’s question and attempt to synthesize a coherent answer by aggregating the relevant facts. At the other end of the spectrum, there are data intensive systems [4] that attempt to use the redundancy of the web to arrive at an answer for factoid style questions. There are also variants of such question answering techniques that involve interaction and use context to resolve ambiguity [5]. In contrast to these approaches, our method matches the user’s question against the questions in a large corpus of question-answer pairs and retrieves the corresponding answers.

Another set of related work is from the information retrieval community. In [6], the authors match the user’s question to a frequently-asked-questions (FAQ) database and select the answer whose question matches most closely to the user’s question. An extension of this idea is explored in [7, 8], where the authors match the user’s query to a community collected question-answer archive such as [9, 10]. Our approach is similar to both these lines of work in spirit, but differs in that the user’s query in our system originates as a spoken query, instead of a text query as in previous work. We also address the issue of noisy speech recognition and assess the value of tight integration of speech recognition and search in terms of improving the overall performance of the system.

There is much less research on spoken question answering than on text-based question answering. QUASAR [11] is a spoken QA system in which the spoken query is recognized and keywords from the recognition output are used to do passage retrieval and answer extraction. Dialog Navigator [12] is a spoken QA system where recognition errors of the spoken query are corrected by repeated dialogs with the user, and the corrected string is used to answer the user query. These systems take a serialized approach to spoken QA — ASR followed by QA. In contrast, our approach is to tightly couple ASR and Search, by re-ranking the ASR output using the constraints
(encoded as relevance metrics) from the Search task [17].

Also related is the literature on voice-search applications [13, 14, 15, 16] which provide a spoken language interface to business directories and return the phone number, address and web sites of businesses. The user’s input is typically not a free flowing natural language query and is limited to expressions with a business name and a location. In our system, user’s can avail of the full range of natural language expressions to express their information need.

3. SPEECH-DRIVEN QUESTION RETRIEVAL SYSTEM

We describe the speech-driven query retrieval application in this section. The user of this application provides a spoken language query to a mobile device intending to find an answer to the question. The query is not constrained to be of any specific question type (for example, what, where, when, how). Some example users’ inputs are what is the fastest animal in water, how do I fix a leaky dishwasher, why is the sky blue. The result of the speech recognizer is used to search a large corpus of question-answer pairs to retrieve the answers pertinent to the user query. The result from the speech recognizer can be a single-best string or a weighted word lattice. For this paper, the ASR used to recognize these utterances incorporates an acoustic model adapted to speech collected from mobile devices and a four-gram language model that is built from the corpus of questions. The retrieved results are ranked using metrics discussed in Section 4.1.

Fig. 1. Architecture of the speech-driven question-answering system

4. METHODS OF RETRIEVAL

We formulate the problem as follows. Given a question-answer archive QA = \{ (q_1, a_1), (q_2, a_2), \ldots, (q_N, a_N) \} of N question-answer pairs, and a user’s question u, the task is to retrieve a subset QA^r = \{ (q'_1, a'_1), (q'_2, a'_2), \ldots, (q'_M, a'_M) \} where M << N using a selection function Select and rank the members of QA^r using a scoring function Score such that Score(q_u, (q'_i, a'_i)) > Score(q_u, (q'_i+1, a'_i+1)). For this paper, we assume Score(q_u, (q'_i, a'_i)) = Score(q_u, q'_i).

We have separated the search process into two steps: a selection step and a ranking step. The Select function is intended to select the matching questions that have high “semantic” similarity to the user’s question. However, given there is no objective function that measures semantic similarity, we approximate it using different metrics discussed below.

The ranking of the members of the retrieved set can be based on the scores computed during the selection step or can be independently computed based on other criteria such as popularity of the question, credibility of the source, temporal recency of the answer, geographical proximity of where the answer originated.

4.1. Question Retrieval Metrics

As mentioned before, we retrieve question-answer pairs from the data repository based on the similarity of match between the user’s query and the set of questions in the repository. To measure the similarity, we have experimented with the following metrics.\(^1\)

1. TF-IDF metric: The user input query and the document (d) (in our case, questions in the repository) are represented as bag-of-n-grams (aka terms). The term weights are computed using a combination of term frequency (tf) and inverse document frequency (idf) [18]. If Q = q_1, q_2, \ldots, q_N is a user query, then the aggregated score for a document d using a unigram model of the query and the document is given as in Equation 1. For a given query, the documents with the highest total term weight are presented as retrieved results. Terms can also be defined as n-gram sequences of a query and a document. In our experiments, we have used up to 4-grams as terms to retrieve and rank documents.

\[
Score(d) = \sum_{w \in Q} tf_{w,d} \times idf_w
\]

2. String Comparison Metrics: Since the user query and the query to be retrieved are similar in length, we use string comparison methods such as Levenshtein edit distance [19] and n-gram overlap (BLEU-score) [20] as similarity metrics.

5. TIGHTLY COUPLING ASR AND SEARCH

Most of the speech-driven search systems use the 1-best output from the ASR as the query for the search component. Given that ASR 1-best output is likely to be erroneous, this serialization of the ASR and search components might result in sub-optimal search accuracy. A lattice representation of the ASR output, in particular, a word-confusion network (WCN) transformation of the lattice, compactly encodes the n-best hypothesis with the flexibility of pruning alternatives at each word position. An example of a WCN is shown in Figure 2. The weights on the arcs are to be interpreted as costs and the best path in the WCN is the lowest cost path from the start state (0) to the final state (4). Note that the 1-best path is how old is mama, while the input speech was how old is obama which also is the WCN, but at a higher cost.

Fig. 2. A sample word confusion network with costs on the arcs computed as negative logarithms of the posterior probabilities.

5.1. Representing Search Index as an FST

Lucene [21] is an off-the-shelf search engine that implements the TF-IDF metric. However, we have implemented our own search engine using finite-state transducers (FST) for the following reason. The oracle word/phrase accuracy using n-best hypotheses of an ASR is usually far greater than the 1-best output. However, using each of the n-best (n > 1) hypothesis as a query to the search component is computationally sub-optimal since the strings in the n-best hypotheses usually share large subsequences with each other. The FST representation of the search index allows us to consider lattices/WCNs as input queries.

\(^1\)These metrics are different from the metrics that we used for voice-enabled search in our previous work [17]
The FST search index is built as follows. We index each question-answer (QA) pair \((q_i, a_i), qa_i\) for short) from our repository using the words \((w_{q_i})\) in question \(q_i\). This index is represented as a weighted finite-state transducer (SearchFST) as shown in Figure 3. Here a word \(w_{q_i}\) (e.g. old) is the input symbol for a set of arcs whose output symbol is the index of the QA pairs where old appears in the question. The weight of the arc \(c(w_{q_i}, qa_i)\) is one of the query relevance metrics discussed above. As can be seen from Figure 3, the words how, old, is, and obama contribute a score to the question-answer pair qa25; while other pairs, qa150, qa12, qa450 are scored by only one of these words.

5.2. Search Process using FSTs

A user’s speech query, after speech recognition, is represented as an FSA (either 1-best or WCN), a QueryFSA. The QueryFSA is then transformed into another PSA NgramFSA that represents the set of n-grams of the QueryFSA. The costs from the WCNs are taken into account in building the NgramFSA resulting in a weighted FSA. The NgramFSA is composed with the SearchFST and we obtain all the arcs \((w_q, qa_{w_q}, c(w_q, qa_{w_q}))\) where \(w_q\) is a query term, \(qa_{w_q}\) is a QA with the query term and, \(c(w_q, qa_{w_q})\) is the weight associated with that pair. We aggregate the weight for a QA pair \((qa_{w_q})\) across all query words and rank the retrieved QAs in the descending order of this aggregated weight. We select the top \(N\) QA pairs from this ranked list. The query composition, QA weight aggregation and selection of top \(N\) QA pairs are computed with FST operations as shown in Equations 2 and 3.

\[
D = \pi_2(NgramFSA \circ SearchFST)
\]

\[
TopN = fsmdeterminize(fsmdbestpath(D), N)
\]

The process of retrieving documents using the Levenshtein-based string similarity metric can also be encoded as a composition of FSTs.

6. EXPERIMENTS

We have a fairly large data set consisting of over a million question-answer pairs. The average length of a question in this data set is approximately 10 words, while the average length of an answer is approximately 20 words. More than 30% of the questions in the corpus match exactly with at least one other question in the data set. This statistic lends strong credence to the intuition that people often ask questions that have already been asked before, and supports the development of an automatic question-answering system based on a retrieval mechanism instead of synthesizing an answer.

In order to evaluate the retrieval methods discussed earlier, we segment our test set of QA pairs into a 450 Seen and a 300 Unseen QA sets. The queries in the Seen set have an exact match with some question in the database, while the queries in the Unseen set do not match any question in the database exactly. However, questions in both the Seen and Unseen sets have a human generated answer that is used in our evaluations.  

For each query, we retrieve the twenty most relevant QA pairs, ranked in descending order of the value of the particular metric under consideration. However, depending on whether the user query is a seen or unseen, the evaluation of the relevance of the retrieved question-answer pairs is different as discussed in section 6.1.

6.1. Evaluation Metrics

For the set of Seen queries, we evaluate the relevance of the retrieved top-20 question-answer pairs in two ways:

1. Retrieval Accuracy of Top-N results: We evaluate where in the list of top twenty retrieved questions, does the question that matches the user query exactly is located, whether it is in the top-1, top-5, top-10, top-20 or not in top-20.

2. Coherence of the resulting top-5 result set with respect to the input user query: Coherence attempts to capture the homogeneity of the questions retrieved, with the assumption that the user wants to see similar questions as the returned results. We compute coherence as the mean of the BLEU-score between the input query and the set of top-5 retrieved questions.

For the set of Unseen queries, evaluating the relevance of the retrieved top twenty question-answer pairs for each input user query is less straightforward. There are no questions in the database that exactly match the input query. So, we evaluate the relevance of the top-20 retrieved question-answer pairs in the following way. For each of the 300 Unseen queries, we know the human-generated answer. Now, we compute the BLEU-score between the human-generated answer and each of the answers in the repository to identify the closest answer in the database to the human-generated answer. We refer to this pair as the Best-Matched QA pair for that unseen query. Using the Best-Matched QA pair for each Unseen query, we evaluate where in the list of top twenty retrieved QA pairs, does the Best-Matched QA pair occur, for each retrieval method.

6.2. Results

On the Seen set of queries, as expected the retrieval accuracy scores for the various retrieval techniques performed exceedingly well. The unigram based tf.idf method retrieved 93% of the user’s query in the first position, 97% in one of top-5 positions and 100% in one of top-10 positions. All the other retrieval methods retrieved the user’s query in the first position for all the Seen queries (100% accuracy).

In Table 1, we tabulate the results of the Coherence scores for the top-5 questions retrieved using the different retrieval techniques for the Seen set of queries. As can be seen, the higher the \(n\)-gram the more coherent the set of the results to the user’s query. It is interesting to note that the BLEU-score and Levenshtein similarity driven retrieval methods do not differ significantly in their scores from the \(n\)-gram tf.idf based metrics.

---

2For \(n\)-gram sequences, the FST search index has intermediate states.

3We have dropped the need to convert the weights into the real semiring for aggregation, to simplify the discussion.

4There may however be semantically matching questions.

5The reason it is not a recall and precision curve is that, for the “seen” query set, the retrieval for the questions is a zero/one boolean accuracy. For the “unseen” query set there is no perfect match with the input question in the query database, and so we determine the closeness of the questions based on the closeness of the answers.
### Table 1. Coherence metric results for top-5 queries retrieved using different retrieval techniques.

<table>
<thead>
<tr>
<th>Method</th>
<th>Coherence Metric for top-5 results</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF</td>
<td>unigram: 61.38, bigram: 60.23, trigram: 66.23, +-gram: 89.74</td>
</tr>
<tr>
<td>BLEU-score</td>
<td>66.29</td>
</tr>
<tr>
<td>Levenshtein</td>
<td>67.50</td>
</tr>
</tbody>
</table>

Table 1. Coherence metric results for top-5 queries retrieved using different retrieval techniques.

In Table 2, we present the retrieval results using different methods on the Unseen queries. We consider that a question cannot be answered if its Best-Matched answer is less than 30 BLEU score points from the human-generated reference answer for the query. With this rejection level, we find that 57% of the Unseen queries cannot be answered using the database. For the 43% queries that are answered, it is interesting to note that over 40% have their Best-Matched question-answer pair retrieved in the top-1 position. We expect the coverage to improve considerably by increasing the size of the QA archive.

### Table 2. Retrieval results for the Unseen queries

<table>
<thead>
<tr>
<th>Method</th>
<th>Top-1</th>
<th>Beyond Top-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF</td>
<td>Unigram: 44.5, Bigram: 46.1, Trigram: 45.3, +-gram: 44.5</td>
<td></td>
</tr>
<tr>
<td>BLEU-score</td>
<td>42.2</td>
<td>37.0</td>
</tr>
<tr>
<td>Levenshtein</td>
<td>41.4</td>
<td>37.0</td>
</tr>
</tbody>
</table>

Table 2. Retrieval results for the Unseen queries

#### 6.3. Speech-driven query retrieval

In Equation 4, we show the tight integration of WCNs and SearchFST using the FST composition operation (◦). \( \lambda \) is used to scale the weights\(^6\) from the acoustic/language models on the WCNs against the weights on the SearchFST. As before, we use Equation 3 to retrieve the top \( N \) QA pairs. The tight integration is expected to improve both the ASR and Search accuracies by co-constraining both components.

\[
D = \pi_2(\text{Unigrams}(WCN) \lambda \circ \text{SearchFST})
\]  

(4)

We use a different set of 895 speech utterances split into development (250 utterances) and test (645 utterances) sets. In Table 3, we show the Word and Sentence Accuracy measures for the best path in the WCN before and after the composition of SearchFST with the WCN on the development and test sets. We note that by integrating the constraints from the search index, the ASR accuracies can be improved by about 1% absolute.

### Table 3. ASR accuracies of the best path before and after (in parenthesis) the composition of SearchFST

<table>
<thead>
<tr>
<th>Set</th>
<th># of utterances</th>
<th>Word Accuracy</th>
<th>Sentence Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dev Set</td>
<td>250</td>
<td>77.1(78.2)</td>
<td>54(54)</td>
</tr>
<tr>
<td>Test Set</td>
<td>645</td>
<td>70.8(72.1)</td>
<td>36.7(37.1)</td>
</tr>
</tbody>
</table>

Table 3. ASR accuracies of the best path before and after (in parenthesis) the composition of SearchFST

In Table 4, we show the search results on 1-best and the WCN. As can be seen, the integration of the ASR WCNs with the SearchFST improves the overall search accuracy as well. The effect of co-constraining the speech and search components is more apparent on top-1 result, as expected.

### Table 4. Search accuracies on 1-best and WCNs.

<table>
<thead>
<tr>
<th></th>
<th>Top-1</th>
<th>Top-3</th>
<th>Top-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-best</td>
<td>47.4</td>
<td>51.3</td>
<td>53.1</td>
</tr>
<tr>
<td>WCN</td>
<td>48.3</td>
<td>51.3</td>
<td>53.1</td>
</tr>
</tbody>
</table>

Table 4. Search accuracies on 1-best and WCNs.

7. CONCLUSION

We have presented a system for using spoken language queries to retrieve questions from a large question-answer archive. We have also presented results that not only evaluate the effectiveness of different retrieval methods, but also measure the coherence of the retrieved set of questions relative to the user’s query. Furthermore, we have shown improvements in speech recognition accuracy by tightly coupling the speech recognition and search models.

8. REFERENCES