Mobile search is a fast-growing business. Mobile voice search provides an easier way to search for information from mobile devices using voice. Natural language understanding (NLU) is a key component technology in voice search to assure search effectiveness. This paper describes a general framework for building the NLU modules in voice search applications. The NLU task is defined as segmenting ASR output, including ASR 1-Best and ASR Word Confusion Networks, into several concepts that are necessary for high-precision search. Application data such as raw query logs, annotated queries and source database are used to train the NLU models. We instantiated this framework on a mobile business search application and demonstrated the flexibility of using this framework. We report the experimental results on this application.

Index Terms—Voice Search, Natural Language Understanding

1. INTRODUCTION

Voice search is an integration of automatic speech recognition (ASR) and text or database search. In this paper, we propose a Natural Language Understanding (NLU) module between ASR and Search as shown in Figure 1. As expected the ASR and Search components perform speech recognition and search tasks. The role of NLU is twofold: (a) parse the automatic speech recognition (ASR) output (1-best and word lattices) into meaningful segments that contribute to high-precision search, and (b) understand user’s intent. This paper focuses on the first task. There are a number of motivation factors for why a NLU module is needed in mobile voice search.

First, one of the main challenges in mobile voice search is the mismatch between error-prone ASR in the presence of ambient noise and high user expectation of search effectiveness. Suppose a voice query free Wi-fi restaurant is misrecognized as free five restaurant. Search would not satisfy users’ needs unless the system can recover the words Wi-fi. It is likely, however, the word Wi-fi is included in ASR lattice output though it is not part of the ASR 1-best string. Hence, as shown in Figure 1, we propose the NLU module takes as input not only ASR 1-best and also ASR Lattices as input for the goal of improving search robustness to ASR errors. The hope is that NLU favors wi-fi over five in this example.

Second, the information users search by and search for is not limited to text only. The similarity measure between user query and relevant targets is beyond keyword match. For instance, business search, a popular mobile search application, searches through a database of many fields for businesses of user interest. A user might query for restaurants near Florham Park. Ranking of restaurants should be based on their distance to Florham Park, instead of keyword overlapping with Florham Park. One role of the NLU module is to segment the input query into segments and associate each segment with needed concepts in the application. In [1], we described a probabilistic query parsing approach using text indexing and search. The task was to segment a natural language query into two fields, namely SearchTerm and LocationTerm, for local business search. In this paper, we extend the work in [1] to be a generic NLU framework for voice search.

Third, mobile voice search needs to achieve high performance level for user acceptance. Even when all database fields are text and a database record can be treated as a concatenated ad-hoc document, [2] showed a field sensitive model significantly outperformed ad-hoc search when searching multi-field media data.

Lastly, users tend to be more verbose when naturally speaking a request. For example, instead of phrasing the query as best restaurants near Florham Park, users may utter I want the best restaurants nearby. The NLU module here is needed to filter out carry phrases and translate nearby to the device location.

There are several threads of relevant research literature including spoken language understanding in dialog applications, query segmentation, named entity extraction, as well as natural language interface to database. We provided an overview of related work in [1].
In this paper, we present a general framework for building NLU modules from application data. The paper outline is as follows. In Section 2, we describe our proposed framework in detail including the problem formulation, training approaches, and implementation. Section 3 reports the experimental results. We summarize our results in Section 4.

2. A GENERAL NLU FRAMEWORK TO VOICE QUERY PARSING

In this section, we formulate the NLU task, namely, parsing ASR output (1-best and word lattices) into meaningful segments that contribute to high-precision voice search, as follows.

**Input**: The NLU module takes ASR lattices in the form of Word confusion networks (WCNs), represented as \( Q_{wen} \), as input. WCN is a compact lattice format [3]. It aligns a speech lattice with its top-1 hypothesis, yielding a sausage-like approximation of lattices. It has been used in applications such as word spotting and spoken document retrieval. Figure 2 shows an example of WCN. There are one or multiple arcs between a pair of consecutive nodes. Symbols on these arcs are alternative words for the given word position. For instance, on Figure 2, gary, cherry, dairy, jerry are possible choices for the first word. Numbers on the arcs are negative log posterior probabilities of the associated word. ASR 1-best is a special case of WCN, where there is only one word for each word position and posterior probabilities are uniformized.

**NLU Parsing**: The NLU task is to segment \( Q_{wen} = q_1, q_2, \ldots, q_i, \ldots, q_n \) into a sequence of concepts. Each \( q_i \) is a set of possible words on the arcs of the \( i \)th word position \( q_i = \{ w_{a(i)} \mid 1 \leq a(i) \leq na_i \} \), where \( na_i \) is the number of available arcs. Each concept can possibly span multiple words. Let \( W^j_i = w_{a(i),1}, \ldots, w_{a(i),j} \) be one possible word sequence from the \( i \)th word to the \( j \)th word. \( a(i) \) and \( a(j) \) are indices of the arcs. Let \( S = s_1, s_2, \ldots, s_k, \ldots, s_m \) be one of the possible segmentations comprising of \( m \) segments, where \( s_k = W^j_i \). The corresponding concept sequence is represented as \( C = c_1, c_2, \ldots, c_k, \ldots, c_m \).

\[
(S^*, C^*)|Q_{wen} = \arg\max_{\{S,C|Q_{wen}\}} P(S, C) \cdot P_C(S)^{\lambda_c} (1)
\]

For a given \( Q_{wen} \), we are interested in searching for the best segmentation and concept sequence \((S^*, C^*)\) as defined by Equation 1, which is rewritten by Bayes rule as Equation 2 with an extra parameter \( \lambda_c \). There are three components in Equation 2. \( P(C) \) is the prior probability of the concept sequence. We use \( \lambda_c \) to scale the prior \( P(C) \). \( P(S|C) \) is the segment sequence generation probability. \( P_{cf}(S) \) is the posterior probability of the word sequence of \( S \) on \( Q_{wen} \). \( \lambda_{cf} \) is used to adjust the influence of ASR posterior probabilities.

The values of both \( \lambda_c \) and \( \lambda_{cf} \) are determined empirically. We will describe how these probabilities are learned from data later in this section.

Beyond optimizing NLU for query segmentation and concept assignment, we are interested in bringing the goal of NLU closer to the search effectiveness. It is reasonable to assume that search only returns good relevant results when the query is meaningful. For instance, if an input query is only the word open, search will not return anything considered as relevant. Another related assumption is that users tend to post meaningful queries. Hence, in [1], we proposed to re-rank ASR WCNs to prefer paths containing a query subject. We defined a query subject as the core concept of the query, which is the must match part. Each valid query has a query subject. For examples, night club is the query subject in night clubs open christmas day. “wi-fi” is the query subject in a query wi-fi. We represent the query subject probability as \( P_{sb}(S) \) and introduce it as the forth component to the NLU optimization. Equation 2 hence is extended to Equation 3.

\[
(S^*, C^*)|Q_{wen} = \arg\max_{\{S,C|Q_{wen}\}} P(S, C) \cdot P(C)^{\lambda_c} \cdot P_{cf}(S)^{\lambda_{cf}} \cdot P_{sb}(S)^{\lambda_{sb}} (3)
\]

Where \( \lambda_{sb} \), similar to \( \lambda_{cf} \), is a scaling factor to the query subject probability.

**Training**: There are four probabilities in Equation 3. ASR posterior probabilities \( P_{cf}(S) \) are retrieved from a given \( Q_{wen} \). We model \( P(C) \), \( P(S|C) \) and \( P_{sb}(S) \) from the application data.

We approximate the prior probability \( P(C) \) using an n-gram model on the concept sequence as shown in Equation 4. Training examples of concept sequences can be created from annotated queries.

We model the segment sequence generation probability \( P(S|C) \) as shown in Equation 5, using independence assumptions. The query terms corresponding to a segment and concept are generated using Equation 6. A corpus of instantiations of the concept \( c_k \) is needed to infer conditional probabilities \( P(W^j_i | c_k) \). This corpus can be a union of query logs, database field values and human generated examples. One way of approximating \( P(W^j_i | c_k) \) is to use relative frequency as defined in Equation 7, where \( d_k \) is a corpus of examples collected for the concept \( c_k \); \( tf(W^j_i, d_k) \) is the term frequency, the frequency of \( W^j_i \) in \( d_k \); \( N \) is the number of entries in \( d_k \); \( \sigma \) is an empirically determined smoothing factor.

\[
P(C) = P(c_1) \prod_{k=1}^{m} P(c_k | c_{k-1}, \ldots c_{k-h+1}) (4)
\]

\[
P(S|C) = \prod_{k=1}^{m} P(s_k | c_k) (5)
\]

\[
P(s_k | c_k) = P(W^j_i | c_k) (6)
\]
We estimated the query subject probability $P_{sb}(W_j^i)$ through mining query logs, which latently encapsulate the most likely subject phrases. Subject phrases are phrases appearing often as a complete query. For instance, the term night club is one of the frequent independent queries. However, we rarely see open Christmas Day as an independent query, though it is a frequent phrase. It often co-exists with other keywords or phrases as a constraint to the main concept in the query. Based on these heuristics, we proposed in [1] to define the query subject probability as the likelihood of a phrase $S$ to be a complete query or an independent concept in the given application. $P_{sb}(W_j^i)$ is formulated in Equation 8 and Equation 9, where $Freq(s)$ is the number of times that $s$ appears as a complete query in the corpus. $N$ is the number of entries in the corpus and $\gamma = 1/nw(s)$. We use $\gamma$ to prefer longer phrases as the subject of the query. We create this corpus by concatenating simple queries in the query log and values of certain fields in the database. Simple queries are defined as queries that do not have explicit constraint indicators such as that, with, etc. or contain less than a given number of words. Each simple query or database field value is treated as an entry in the corpus.

$$P(W_j^i | c_k) = \frac{tf(W_j^i, d_k) + \sigma}{N(1 + \sigma)}$$ (7)

$$P_{sb}(S) = \max_{s \in \{s_1, ..., s_m\}} P_{sb}(s)$$ (8)

$$P_{sb}(s) = (\lambda \ast \frac{Freq(s)}{N} + (1 - \lambda) \ast \frac{1}{N})^\gamma$$ (9)

**Framework** We illustrate the training and parsing framework described above in Figure 3. Training data include query logs, application database and annotated queries. Three probability models, namely, the prior probability $P(C)$, the concept generation probability $P(s|c_k)$ and the query subject probability $P_{sb}(S)$, are learned from the data. The NLU parser takes as input ASR WCNs and the NLU models. It outputs parsing results in the form of Concept-Value pairs. For instance, in business search, the NLU parses night club in New York into: Category: night club and City: New York.

**Implementation** As we discussed in [1], there is a lot of related work and many potential approaches such as those for NE extraction we can explore to implement this framework.
abilities in Equation 3. Two of them, \( P_{cf}(S) \) and \( P(C) \), are weighted finite-state acceptors (FSA). \( P_{cf} \) is from \( Q_{wcn} \), which is encoded by ASR as a FSA. We also represent the concept n-gram model for the prior \( P(C) \) as a FSA. We create a finite state transducer (FST) \( G \) from \( Q_{wcn} \) to include all possible ways of segmenting the input query, where input symbols are segments \( s \) and output symbols are concepts. Weights on the FST are \(-\log(P(s|c_k)) \cdot P_{sb}(s|\lambda_{s_k}) \cdot P_{cf}(s|\lambda_{c_f})\), a combination of the segment generation probabilities, the subject probabilities and the WCN posterior probabilities. The parsing task of finding \((S^*, C^*)\) is then a search for the lowest weight path of a composed FST from \( C \) and \( G \) as shown in Equation 10. The input and output symbol sequence \((\pi_2)\) from the lowest weight path is the pair \((S^*, T^*)\).

\[
(S^*, C^*) = \pi_2(\text{Bestpath}(G \circ C^{\lambda c})) \quad (10)
\]

3. EXPERIMENTS

We instantiated the described framework on a mobile local business search task, where the search engine we use is [http://www.yellowpages.com](http://www.yellowpages.com) and the speech recognizer is the AT&T Watson ASR engine [6]. Our training data consists of 18 million web queries to [http://www.yellowpages.com](http://www.yellowpages.com), where a query comprises two fields, SearchTerm and LocationTerm, 11 million unique business entries, and 15 thousand annotated voice queries. The NLU task is to parse a voice query into three fields, namely SearchTerm, LocationTerm and Filler. We tested our approaches on 1000 randomly selected voice queries from a newer time period than the training data.

We used an ASR with a trigram-based language model trained on web query logs and achieved 67.2\% ASR word accuracy and 55.6\% sentence accuracy. We measure the NLU parsing performance in terms of extraction accuracy on the two non-filler slots: SearchTerm and LocationTerm. Extraction accuracy computes the percentage of the test set where the string identified by the parser for a slot is exactly the same as the annotated string for that slot.

Table 1 reports the parsing performance for the two slots. The **Transcription** column presents the parser’s performances on human transcriptions (i.e. word accuracy=100\%) of the speech. The **1-best** and WCN respectively corresponds to ASR 1-best and WCN output. As expected, the parser’s performance is much lower for ASR output than on human transcription. Though 63.0\% SearchTerm extraction accuracy on ASR output is low, search performance is much higher for its robustness to ASR errors such as **restaurant** being misrecognized as **restaurants**. The promising aspect is that we improved SearchTerm extraction accuracy by 2.0\% when using WCN as input. LocationTerm extraction accuracy remains almost unchanged (0.1\% drop), since the query subject model applies mainly on the SearchTerm segments.

<table>
<thead>
<tr>
<th>Slots</th>
<th>1-best</th>
<th>WCN</th>
<th>Transcription</th>
</tr>
</thead>
<tbody>
<tr>
<td>SearchTerm</td>
<td>61.0%</td>
<td><strong>63.0%</strong></td>
<td>95.1%</td>
</tr>
<tr>
<td>LocationTerm</td>
<td>88.5%</td>
<td>88.4%</td>
<td><strong>97.4%</strong></td>
</tr>
</tbody>
</table>

Table 1. Slot Extraction Accuracy of the NLU Parser

4. SUMMARY

This paper described a general framework for building the NLU modules in voice search. The task is to parse ASR output into the needed concepts of the application. We described an approach to training the NLU models including concept prior probability, query segment generation probability, and query subject probability from application data such as query log and source database. We presented an implementation of this framework using a generic search engine and FSM tools. We built a NLU module for a business search application using this framework and reported the performance in Section 3. The near future work is to apply this framework on other voice search applications when data becomes available.

5. ACKNOWLEDGEMENTS

We are grateful to Mazin Gilbert, Srinivas Bangalore, Diamantino Caseiro, Vincent Goffin, and Barbara B. Hollister for their help in discussing and improving the ideas presented in this paper.

6. REFERENCES


