MULTIPASS STRATEGIES FOR IMPROVING ACCURACY IN A VOICE SEARCH APPLICATION

1,2 Tianhe Zhang, 2 Richard Rose, and 1 Jean Dahan

1 Nuance Communications 2 Department of Electrical and Computer Engineering, McGill University Montreal, Quebec, Canada

ABSTRACT
This paper describes a set of techniques for improving the performance of automated voice search services intended for mobile users accessing these services over a range of portable devices. Voice search is implemented as a two stage search procedure where string candidates generated by an automatic speech recognition (ASR) system are re-scored in order to identify the best matching entry from a potentially very large application specific database. The work in this paper deals specifically with user utterances that contain spoken letter sequences corresponding to spelled instances of search terms. Methods are investigated for identifying the most likely database entry associated with the decoded utterance. An experimental study is presented describing the characteristics of actual user utterances obtained from a prototype voice search service. The impact of these methods on word error rate is presented.

Index Terms—Speech recognition, String matching

1. INTRODUCTION

There are a large number of tasks involving access to online information that are easy to perform on a workstation with a keyboard and large display but are more difficult to perform on a mobile device with more limited input and display modalities. This paper addresses the problem of making the functionality associated with online text search engines accessible to users of a wide range of mobile devices. Specifically, techniques for improving the performance of a prototype service allowing users to enter query terms to an online search engine by voice are presented and evaluated.

These techniques are investigated in the context of an existing implementation of a voice search service. After accessing the service, mobile users speak one of a set of 5 category names, for example “business names”, and then speak the query term for that category. The utterance is presented to an automated speech recognition (ASR) system and the decoded string is then presented as a text based query string to a database that may contain anywhere from approximately 10,000 to over one million entries depending on the search category. Utterances where an ASR based confidence measure does not exceed a threshold are directed to a human operator. Analysis of 3900 utterances collected from users’ interactions with this service has provided the motivation for the approaches described in Section 3 in the paper. Section 2 describes this analysis which revealed that many user utterances contain spoken letter sequences. This is characteristic of customer interaction with human directory assistance operators where names are often spelled as a means of clarification.

Section 3 describes multipass methods for identifying the most likely database entry associated with the utterances that contain spoken letters. A combined grapheme and phoneme based similarity measure is used to match decoded n-best letter string hypotheses obtained from the ASR system with entries in the database. Discriminative model combination (DMC) is used to obtain an optimum linear combination of the log probabilities used in rescoring [1]. Finally, the results of an experimental study performed using a set of 8000 utterances collected from users of the service are presented in Section 4.

2. BACKGROUND AND MOTIVATION

This section briefly introduces the voice search task domain and motivates the approaches investigated in this work. The voice search task being addressed here involves spoken queries entered by users of mobile devices. Users perform directed queries by speaking one of a set of 5 category names at the beginning of an utterance. Examples of search categories include business names, stock quotes, and website names. A pilot corpus containing 3935 labeled utterances has been collected from a population of 601 users of the service. While the utterances begin with a search category name, they are otherwise relatively unconstrained. Analysis of the data shows that there are approximately seven words per utterance. Approximately 37 percent of the utterances contain spoken letter sequences which correspond to “spell mode” versions of a query term.

This large number of spell mode utterances is problematic because the WER obtained for utterances containing spoken letters was found to be approximately 8 percent higher than the WER for utterances that did not contain spoken letters.
3. MULTIPLE PASS SPOKEN TERM RETRIEVAL

This section describes the approach for detecting query terms in a potentially very large text database from utterances containing spoken letter sequences corresponding to these terms. It is based on the multipass search algorithm summarized in the block diagram shown in Figure 1. An ASR system produces a list of the n most likely hypothesized word sequences. A spoken letter sequence, if any, is extracted from each candidate n-best sequence and the letter sequence is rescored with respect to both grapheme and phoneme expansions of the entries in the text database. The techniques used for rescoring the candidate letter sequences are described below.

![Diagram](Image)

**Fig. 1.** Rescoring n-best ASR candidates obtained from spell mode utterances

### 3.1. Rescoring ASR hypotheses

There are several procedures that are presented here for implementing this re-scoring process. It is assumed that the hypothesized letter sequences produced by the ASR system may contain either recognition errors produced by the ASR system or the query term may have been misspelled by the speaker. There are two constraints applied to deal with these potential errors. In the case of recognition errors, it is assumed that the letter sequence may contain simple letter substitutions, insertions, and deletions. To deal with this class of errors, a dynamic programming based string alignment of the hypothesized letter sequence with respect to graphemic expansions of the entries in the database is performed. This process yields a set of grapheme based distances, \( S_g(n, m) \), between the \( n \)-th ASR candidate and the \( m \)-th database entry as shown in Figure 1. Because of the potentially very large number of database entries, it is important that an efficient means is used for computing these scores. This will be described in Section 3.3.

In the case of the utterances themselves containing mis-spellings, it is assumed that the hypothesized letter sequence corresponds to a word or words that “sounds similar” to the correct spelling. To deal with this case, a phonemic expansion of the hypothesized grapheme sequence is obtained. Then a DP based alignment of the resulting phoneme sequence with respect to phonemic expansions of the database entries is performed [4]. This process yields a set of phoneme based distances, \( S_p(n, m) \), shown in Figure 1. The phonemic expansions are obtained using a statistical model that obtains the phone sequence, \( p = p_1, p_2, \ldots, p_L \), associated with an N length sequence of graphemes, \( g = g_1, g_2, \ldots, g_M \), that maximizes the likelihood

\[
\hat{p}(g) = \arg\max_p p(p, q)
\]  

(1)

The model is based on representing the joint grapheme / phoneme sequence as a sequence of “graphonomic” units [5] where each unit corresponds to a pair of grapheme/phoneme sequences, \( q = (g, p) \).

Additional letter sequence constraints are applied in the form of the log probabilities associated with a letter based statistical language model (LM). This is done by training 3-gram probabilities for letter sequences by first obtaining word transcriptions taken from the domain of a given search category and then expanding the words as grapheme sequences. Though not shown in Figure 1, the LM probabilities are included in the criterion that is used to rescore the ASR n-best list and to obtain the optimum database entry, \( \hat{m}(n) \). This criterion is described in more detail in Section 3.2.

### 3.2. Discriminative Model Combination

For each of the n-best spoken letter string candidates generated by the ASR system, the rescoring procedure described in Section 3.1 finds an optimum database entry, \( \hat{m}(n) \), that optimizes a criterion based on a linear combination of three measures. These include grapheme alignment...
score, \( S_g(n) = S_g(n, \hat{m}) \), phoneme based alignment score, \( S_p(n) = S_p(n, \hat{m}) \), and the letter sequence language model probability, \( p(Y^n_m) \). The alignment scores can be normalized and approximated as the probabilities \( p(X^n_g|Y^n_m) \) and \( p(X^n_p|Y^n_m) \) respectively. The overall optimization criterion is given in terms of the weight vector \( \Lambda = \{\lambda_1, \lambda_2, \lambda_3\} \) as

\[
\lambda_1 \log p(Y^n_m) + \lambda_2 \log p(X^n_p|Y^n_m) + \lambda_3 \log p(X^n_g|Y^n_m). \tag{2}
\]

These weights are estimated discriminatively by minimizing a continuous function of the string error generated by the ASR system on a held out training set \cite{1}.

### 3.3. Efficient Grapheme/Phoneme Search

This section describes the dynamic programming based search algorithm for finding the database entry that is “closest” to a given n-best spell mode recognition hypothesis. While the discussion describes search based on grapheme alignment, the same procedure is used for aligning hypothesized and reference phoneme strings as well.

There are \( N \) hypothesized letter or grapheme sequences generated by the ASR system. The \( n \)th hypothesized sequence is given as \( X^n_g = \{x^n_{1,n}, x^n_{2,n}, \ldots, x^n_{i,n}\} \) where \( x^n_{i,n} \) is the \( i \)th letter or grapheme in the sequence and \( I_m \) is the length of the \( n \)th sequence. For each search category there are \( M \) sequences in the category specific database.

The distance between the grapheme sequence hypothesis, \( X^n_g \) and \( Y^n_m \) is given by \( S_g(n, m) \) and corresponds to a Levenshtein distance computed using a dynamic programming (DP) algorithm. The distance, \( L_{n,m}(i, j) \), between \( i \) length subsequence of \( X^n_g \) and \( j \) length subsequence of \( Y^n_m \) is computed by induction from subsequences of lengths \( i - 1 \) and \( j - 1 \) as:

\[
L(i, j) = \max \left\{ L(i - 1, j) \cdot P(\phi, x_i), L(i - 1, j - 1) \cdot P(y_j, x_i), L(i, j - 1) \cdot P(y_j, \phi) \right\} \tag{3}
\]

where the reference and test sequence indices have been dropped to simplify notation. In Equation 3, \( P(h, r) \) is the probability of the hypothesized letter index \( h \) being confused with the reference letter index \( r \), \( P(h, \phi) \) refers to insertion probability, and \( P(\phi, r) \) is the probability of deletion. These probabilities are estimated from counts computed over training data:

\[
P(h, r) = \frac{C(h, r)}{\sum_{k \neq h} C(k, r)} \tag{4}
\]

where \( C(h, r) \) refers to the number of occurrences of the hypothesized letter \( h \) being aligned with the reference letter \( r \), in held-out training data.

The distance between input string candidate \( X^n_g \) and database entry \( Y^n_m \) can be written in terms of the DP score as \( S_g(n, m) = L_{n,m}(I_n, J_m) \). The computation associated with evaluating \( S_g(n, m) \) for all \( m = 1, \ldots, M \) database entries can be prohibitive to real time operation if the input string \( X^n_g \) is aligned to one reference string at a time as suggested by Equation 3. The computational load associated with computing \( S_g(n, m) \) can be reduced by pruning partial path matches so that the induction in Equation 3 is not performed for subsequences terminating at letter pairs \((i, j)\) when the partial path score \( L_{n,m}(i, j) \) falls below a threshold. A beam search strategy can be implemented where a pruning threshold, \( T(i) \), is obtained by applying a fixed pruning beam width, \( BW \), to the maximum alignment score obtained for the input subsequence of length \( i \), where \( T(i) = \max_j (L(i, j)) - BW \).

A more efficient strategy can be used for computing the distances \( S_g(n, m) \) for input string \( X_n \) in a way that facilitates a more aggressive pruning strategy using an adaptive pruning threshold. The \( i - 1 \) step of this procedure involves aligning all reference strings \( Y^n_m, m = 1, \ldots, M \) to an \( i - 1 \) length substring of \( X^n_g \). Low scoring partial alignments are pruned and all surviving alignments are extended to a length \( i \) length substring of \( X^n_g \) for all \( M \) reference strings using the induction in Equation 3. This is continued until \( i = I_n \). The important aspect of this procedure is that the pruning beam width can be adapted at each step based on information derived from the partial alignment obtained from all reference strings to the \( i \)th input substring.

At the \( i \)th step of the above procedure, a dynamic pruning beam width \( BW(i) \) is obtained from the beam width at step \( i - 1 \) as:

\[
BW(i) = BW(i - 1) \ast s(i) \tag{5}
\]

where \( s(i) \) is a scale factor that has the effect of reducing the beam width as the length of the input substring increases. The scale factor that is applied to \( BW(i) \) is itself dependent on a minimum threshold applied to the value of the beam width, \( BW_{\text{min}} \), and to a minimum threshold, \( NP_{\text{min}} \), applied to the number of active substring alignments:

\[
s(i) = \begin{cases} 
\alpha s(i - 1) & BW(i) > BW_{\text{min}}, NP > NP_{\text{min}} \\
1.0 & \text{otherwise}
\end{cases} \tag{6}
\]

The value of \( s(i) \) is initialized to 1.0 and a value of \( \alpha = 0.9 \) was found to provide a good trade-off between performance and efficiency. Equations 5 and 6 have the effect of continually decreasing the beam width for successively longer input subsequences.

### 4. EXPERIMENTAL STUDY

An experimental study was performed to evaluate the effect of the approaches described in Section 3 on ASR performance for spell mode utterances. The data set consisted of a total of 8000 test utterances collected from 735 speakers performing voice search under one of the three categories described in Section 2. These include 2000 utterances in the “website” category, 4000 in the “stock quote” category, and 2000 in the “business name” category. Confusion matrices were trained by aligning decoded and reference letter and
phoneme sequences on held out training utterances. Separate databases were compiled for the three search categories. The database for the “website” category was originally defined as the names of the top one million websites as measured by user tracking information provided by Alexa [6]. The database for the “stock quote” category consists of 9500 entries originally taken from the NYSE, AMEX, and NASDAQ exchanges. Finally, the database for the “business names” category consists of approximately 40,000 entries acquired from various corpora and augmented by customer searches in this category.

Table 1 presents the performance of the techniques described in Section 3 as the relative reduction in WER for spoken letter sequences with respect to the baseline WER obtained for spell mode utterances in the ASR system of Figure 1. The number of candidate sequences in the n-best list is $n = 10$. The actual baseline WER is not given here. WER reduction is presented separately in each of the three columns of the table for the three different search categories.

There are several observations that can be made from Table 1. First, it is clear from the first row of the table that the grapheme based rescoring (GRC) procedure presented in Section 3 results in an average relative WER reduction of approximately 20 percent. The effect of this procedure is most pronounced for the “website” category. Second, the second row of the table shows that the combination of pronunciation based rescoring (PRC) and GRC results in a small additional reduction in WER. This effect is less pronounced for the stock quote and website categories since many of the query terms in these categories are acronyms or unusual proper names. The last observation relates to the inclusion of the ASR language model (LM) scores in the second pass rescoring of the ASR hypotheses. The third row of Table 1 shows that the letter based language model (LM) in second pass rescoring had a significant impact on the “stock quote” and “business name” search categories.

<table>
<thead>
<tr>
<th>Rescoring</th>
<th>Stock Quote</th>
<th>Websites</th>
<th>Bus. Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRC</td>
<td>18.3%</td>
<td>24.1%</td>
<td>17.2%</td>
</tr>
<tr>
<td>GRC+PRC</td>
<td>19.5%</td>
<td>25.2%</td>
<td>21.0%</td>
</tr>
<tr>
<td>GRC+PRC+LM</td>
<td>22.6%</td>
<td>25.9%</td>
<td>23.0%</td>
</tr>
</tbody>
</table>

Table 1. Relative reduction in WER obtained using grapheme based rescoring (GRC), pronunciation based rescoring (PRC), and language model (LM) scores.

Figure 2 provides a comparison of the execution times and WER reduction associated with the static beam pruning (SBP) and dynamic beam pruning (DBP) DP search strategies described in Section 3. The two curves show the relative WER reduction and the execution time as a percentage of real time (%RT) for the GRC+PRC+LM approach applied to the “website” search category described above. The WER improvements and the execution times represent averages computed over utterances containing letter sequences of length greater than five letters. It is clear from the figure that the DBP strategy provides significantly higher improvements in WER over a range of execution times (%RT=[0.1,0.4]) where the WER improvement has reached an asymptote.

5. SUMMARY AND CONCLUSIONS

This paper has presented techniques for improving the performance of a service providing voice search capability on mobile devices. These techniques were evaluated using utterances collected from users of an actual voice search service. It was observed that in over a third of the interactions users uttered spoken letter sequences as part of their query. The multipass procedures presented in the paper were shown to have the effect of reducing WER by as much as 26.9% and could be efficiently implemented even when the database being searched contained over a million entries. This study provides a good example of how additional domain specific knowledge sources can be used with a domain independent ASR system to facilitate voice access to online search indices.

6. REFERENCES


