AN EFFICIENT APPROACH FOR TWO-STAGE OPEN VOCABULARY SPOKEN TERM DETECTION

Atta Norouzian and Richard Rose

Department of ECE, McGill University, Montreal, Canada

ABSTRACT

This paper investigates indexing strategies for open vocabulary spoken term detection (STD) in a lecture speech domain. STD is performed from word lattices generated offline using an automatic speech recognition (ASR) system configured from a meetings task domain. Indexing of lattice paths is performed to avoid exhaustive search of audio segments which can be impractical for extremely large media repositories. The method is based on constructing a word-based index from these lattices and using an approximate subword-based algorithm for accessing index entries from subword expansions of query terms. Results are presented for an experimental study demonstrating both STD performance and the potential for scaling the indexing strategy to very large collections of audio segments.

Index Terms—Speech recognition, spoken term detection

1. INTRODUCTION

There are many multimedia search applications that require a capability for real-time search of audio documents from query terms entered by a user. These applications are driven by the existence of large media repositories containing recorded lectures, news broadcasts, multi-media messaging services, and conversational telephone speech [1, 2, 3, 4, 5, 6]. There are many human interface scenarios proposed for performing search, organizing search results, and presenting results to users. The scenarios that are anticipated in this work involve queries entered by the user either through a keyboard or by voice with retrieved audio segments returned by the system for user review [1, 2, 7, 5, 6]. All search applications of this type require search to be performed with subsecond response latencies for completely unrestricted user queries even for collections containing hundreds of hours of speech.

The need for extremely fast search while maintaining good spoken term detection (STD) performance for open search vocabularies has dictated a multi-stage approach to the problem. Most successful approaches involve using a speech-to-text system for producing hypothesized word or subword lattices from audio segments followed by a faster-than-real-time search for the desired query term [1, 3, 4, 8, 9]. Typically lattices are generated for audio segments that are less than half a minute in length so it is often considered sufficient to simply determine whether or not a query term has occurred in a given segment. However, even when ASR lattices have been generated in advance, it has been found that approaches requiring exhaustive search of each audio segment do not scale well to very large collections.

To deal with this problem, a number of approaches have been proposed for indexing lattices associated with audio segments [3, 8, 9, 1]. Words or sequences of subwords serve as indexing terms in these indexes and the index is constructed to associate lattices or lattice paths with the indexing terms. Hence, each entry in the index is an indexing term along with a list of lattices or lattice paths associated with that term. Search is performed by identifying the indexing term or terms from the user’s query, accessing the lattices or lattice paths associated with those terms from the index, and performing a detailed search of the lattice paths retrieved from the index.

There are several issues to be considered in constructing indexes of this type. First, the index should be easy to update as new audio material is added to the repository and the index must be of manageable size. Second, the number of entries per indexing term should be small enough to limit the number of segments that are subjected to detailed search but large enough to avoid having many of the actual occurrences of the user’s query being missed. Third, the process of identifying the indexing terms that are relevant to the user’s query must be very efficient. While this last point may seem obvious, allowing for approximate matches between the query and the indexing term can result in significant computational complexity relative to standard text based information retrieval applications.

An open vocabulary spoken term detection approach based on a variant of the above multi-pass search scenario is presented here and evaluated on utterances taken from a lecture speech domain. A set of word lattices are generated off-line for each audio segment using a large vocabulary continuous speech recognition (LVCSR) system. A word based index is constructed where the indexing terms correspond to the set of words in the LVCSR vocabulary. Each entry in the index contains a list of lattice paths that are likely to
contain the vocabulary word. A first pass search procedure is performed to identify lattice paths and the associated audio segments that are likely to contain the query term. Finally, a second pass detailed search based on a constrained and weighted phonemic distance is used to verify the occurrence of the query term in the retrieved audio segments.

This approach will be described in Section 3 where the following three important aspects are emphasized for index construction and search. First, while the index is based on word-based indexing terms, a phoneme based distance measure is used to associate the user’s query with index entries. This facilitates the use of out-of-vocabulary (OOV) query terms by retrieving lattice paths that are likely to contain words that are phonemically “similar” to the query. Second, lattice paths that are likely to contain individual indexing terms are identified during index construction by re-scoring lattices after increasing the prior probability of each indexing term. This results in a smaller set of paths which are likely to be more rich in occurrences of the indexing terms for evaluation during search. Third, the second pass detailed search is performed for audio segments using only a single lattice path associated with that segment rather than the entire lattice. This further reduces the computation associated with search.

The above aspects of the approach makes search extremely efficient while maintaining good STD performance for the lecture speech domain task. Section 4 will describe this task domain and the lecture speech utterances taken from recorded course lectures stored on an online media server [1, 10]. An experimental study performed to evaluate STD performance using the utterances from this domain is presented in Section 5.

2. INDEXING APPROACHES FOR STD

There has been a great deal of work focused on improving detection performance and reducing complexity of spoken term detection (STD) systems. Much of the recent work in this area has been driven by the fact that automatic speech recognition (ASR) performance for application domains that are of interest is often poor. This is a result of several factors including difficult acoustic environments, poor recording quality, and, especially in course lecture scenarios, highly specialized subject matter. This has dictated the need for STD approaches based on ASR lattices rather than simply relying on the single most likely string produced by an ASR system.

To deal with this problem, a number of approaches have been proposed for eliminating the need for exhaustive search of each audio segment. These approaches borrow from the notion of an index in information retrieval (IR) to avoid searching all audio segments for each query by indexing the segments in advance. In IR, an inverted index is used to map a dictionary of indexing terms, generally words, to text documents where those terms have been found to occur. The main challenge of applying these concepts to fast indexing of audio segments is obtaining an alternate definition of index terms and defining how the index terms are associated with the audio segments.

A conceptually straight-forward approach to indexing audio segments is to perform word-based indexing by producing word lattices for each audio segment using a LVCSR system. This was done in [7] where posterior probabilities were estimated for word occurrences in the LVCSR word lattices. An inverted index was constructed for each word in the lexicon. For each word, there is a list of segments ranked by the posterior probabilities estimated for the occurrence of the word in that segment. This has been shown to be an effective approach for in-vocabulary (IV) query terms, but is not applicable for out-of-vocabulary (OOV) queries.

Subword-based strategies have also been investigated for open vocabulary indexing. Phoneme indexes can be created from phoneme lattices generated for audio segments using a subword-based speech recognition system. All the approaches used for locating candidate audio segments for a query term involve obtaining a phonemic expansion for the query term. A procedure for creating a full index containing any possible fixed length phonemic substring of query terms was implemented in [3]. This procedure does not scale well to large collections of audio segments due to the cost of index construction. This issue was addressed in [9] by training a phoneme level n-gram language model (LM) for each segment-based phoneme lattice. The probability of the query string with respect to this LM is used for selecting lattices which are then used for verifying the occurrence of the query. This approach is also difficult to scale to large collections due to the potential size of the index.

Another phoneme-based indexing strategy was proposed in [8]. The indexing terms for this approach are arbitrary phoneme sequences of limited length and each entry in the index is a list of phoneme lattice paths that contain a given phoneme sequence. An algorithm was proposed for identifying these paths to balance the issues associated with index construction and search described in Section 1.

The subword based indexing strategies in [3, 9, 8] represent different trade-offs with respect to the computation required for index construction, index size, and speed of accessing index entries during search. The faster search provided by the index created in [8] represents an advantage in this respect over the strategies used in [3] and [9]. However, it is difficult to compare the detection performance of the different strategies since they are evaluated in different task domains. Performance comparisons made in [9] showed that using separate word based search for IV queries and subword based search for OOV queries yields significantly better performance for IV terms. Hence, this and other studies suggest that there is a performance cost for using unified subword-based indexing for both IV and OOV query terms.
3. HYBRID INDEX FOR TWO-STAGE STD

This section describes a method for fast, open vocabulary STD from ASR word lattices. The method is based on constructing a word-based index from these lattices and using an approximate subword-based algorithm for accessing index entries from subword expansions of query terms. Spoken term detection is performed in three steps. First, a word based index is constructed off-line by identifying lattice paths for each word in the ASR vocabulary. Section 3.1 describes this process. Second, once a query term has been entered by the user, a first pass search is performed to identify audio segments that are likely to contain the query term. Section 3.2 describes how this search is performed for both IV and OOV query terms. The third step, described in Section 3.3, involves verification of the occurrence of the search term through a detailed search of the audio segments identified in first pass search.

3.1. Index Construction

The construction of a word based index begins with a media repository that has been divided into an inventory of audio segments \( S = s_1, \ldots, s_P \) and a set of word lattices, \( \mathcal{L} = l_1, \ldots, l_P \), generated by an LVCSR system for each segment. Given an ASR lexicon, \( \mathcal{V} = \{V_1, \ldots, V_N\} \), the goal is to construct an inverted index so that for each word \( V_i \in \mathcal{V} \), referred to here as indexing terms, there is a list of lattice paths that are likely to contain \( V_i \) and their associated lattices:

\[
V_i : (p_{i,1}, l_{i,1}), (p_{i,2}, l_{i,2}), (p_{i,3}, l_{i,3}), \ldots \tag{1}
\]

In Equation 1, \( p_{i,j} \) is the \( j \)th path for word \( V_i \) and \( l_{i,j} \) is a pointer to the lattice associated with that path. Index construction refers to the process of identifying the paths associated with each \( V_i \).

The criterion used for choosing path \( p_{i,j} \) from lattice \( l_{i,j} \) for indexing term \( V_i \) is that the path have a high probability of containing \( V_i \) relative to other paths in the lattice. Various measures based on lattice posterior probabilities have been defined for either constructing an index or searching for candidate audio segments [7, 3, 8, 9]. However, the approach used here selects a given lattice path for term \( V_i \) in Equation 1 based on a criterion that reflects an expected increase in prior probability for that word relative to the probability predicted by the language model. Candidate paths to be used for verifying the occurrence of a query that is similar to \( V_i \) in lattice \( j \), can be identified by adjusting the prior probability of the occurrence of \( V_i \) for that path.

Modifying the n-gram language model probability for a word in a lattice path can be performed during index construction simply by adding a bias to the path log probability. If the word string for a path is \( \mathbf{W} = w_1, \ldots, w_m \), then the n-gram language model log likelihood is given by \( \mathcal{L}(\mathbf{W}) = \sum_{k=1}^{m} \log P(w_k | w_{k-1}, \ldots, w_{k-n+1}) \). The frequency-based estimate of the probability for word \( w_k \) in the context of its word history is obtained from normalized counts of the word sequence in the training corpus. It is easy to show that scaling these counts for a particular word, \( w_i \), by a multiplicative factor, \( \beta \), is equivalent to adding a constant to the log n-gram probability for that word. For example, in the case of a trigram language model, scaling the tri-gram counts by \( \beta \) for tri-gram probability \( P(w_i | w_{i-1}, w_{i-2}) \) corresponds to

\[
\log P'(w_i | w_{i-1}, w_{i-2}) \approx \log \frac{\beta C(w_i, w_{i-1}, w_{i-2})}{\sum_w C(w_i, w_{i-1}, w_{i-2})} = B + \log P(w_i | w_{i-1}, w_{i-2}), \tag{2}
\]

where the constant \( B \) represents the degree of scaling applied to the word counts in Equation 2. Furthermore, since the total path log likelihood is simply a weighted sum of the acoustic and language log likelihoods, biasing the n-gram log probability as shown in Equation 2 is equivalent to adding \( B \) to the total path log likelihood.

This implies the following strategy for obtaining the list of paths associated with word \( V_i \) shown in Equation 1. For all paths, \( p_i \), in lattice \( j \), increment path likelihoods, \( L_{i,j}' \), for paths containing \( V_i \) by an empirically chosen “boosting factor”, \( B \), to obtain updated path likelihoods

\[
L_{i,j}' = L_{i,j} + mB, \tag{3}
\]

where \( m \) is the number of occurrences of \( V_i \) in \( p_{i,j} \). If any path \( p_i \) becomes the highest ranking path for \( i \) according to \( L_{i,j}' \), then the path and lattice are added as \( (p_{i,j}, L_{i,j}') \) to the index for \( V_i \). Clearly, the number of entries chosen for a given \( V_i \) will depend on both the number of actual occurrences of \( V_i \) and the value of \( B \).

3.2. First Pass Search - Identifying Candidate Paths

Given an input query and its phonemic expansion, \( Q = \{q_1, \ldots, q_n\} \), the first pass search involves identifying the candidate lattice paths that are likely to contain the query term. There are two parts to this process. First, the lattice index terms that most closely match the query are identified. Second, the lattice paths associated with these index terms are retrieved for use in the second pass search for verifying the occurrence of query terms in the candidate audio segments associated with these paths.

Identifying indexing terms, \( V_i \), that are close to the query term, \( Q \), is based on a phonemic distance [1]. A constrained phonemic string alignment is used to compute the distance between the phoneme sequence \( Q \) and the phonemic expansion, \( V_i = \{v_{i,1}, \ldots, v_{i,m_i}\} \), of \( V_i \). The distance between \( Q \) and \( V_i[k] \), the subsequence of \( V_i \) beginning at phoneme index \( k \) is:

\[
\mathcal{M}(Q, V_i[k]) = \frac{1}{n} \sum_{i=0}^{n-1} p(q_i | v_{k+i}). \tag{4}
\]

The probabilities, \( p(q|v) \), in Equation 4 are approximated by normalized counts obtained from phone confusion matrices. These are computed from time aligned decoded and
reference phoneme transcriptions obtained from training utterances taken from the lecture domain.

When a query term, \( Q \), is entered by the user, the matching score for the phoneme string \( Q \) against \( V_j \) is \( \arg \max_k M(Q, V_j[k]) \). The matching scores for all index terms are sorted. The index is then used to obtain the paths associated with the top scoring index terms for use in the second pass detailed search. The set of top scoring index terms for \( Q \) can be represented as \( I_Q \). The impact on detection performance of varying the number of indexing terms, \( |I_Q| \), retained for both IV and OOV query terms is addressed in the experimental study presented in Section 5.

There are two important aspects of this procedure that are worth noting. First, the major impact of the above strategy is that it facilitates the use of a word based index even when the query terms are not contained in the ASR lexicon. Second, using a strategy based on a constrained string alignment for associating query terms with indexing terms is extremely fast. It requires on the order of \(|V|\) string matches of the type shown in Equation 4 for each query, where \(|V|\) is the ASR vocabulary size.

### 3.3. Second Pass Search - Verification

Identifying the candidate lattice paths in first pass search reduces the number of acoustic segments that must be subjected to a detailed search for query term \( Q \). The occurrence of \( Q \) is verified in these segments using a second pass search based on the same phonemic distance given by Equation 4. First, a phonemic expansion is obtained for path \( p_{i,j} \) associated with index entry \( i \in I_Q \) as shown in Equation 1. This phonemic expansion is given by \( H_{i,j} \). Then the phoneme string is aligned with the audio segment, \( s_j \), by Viterbi segmentation with the LVCSR acoustic model. This allows the location of the hypothesized query term occurrence to be reported. Both the phonemic expansion and the Viterbi alignment can be computed during index construction to save computation during search.

The second step in verifying the query term is to compute a score for the phonemic expansion of \( Q \) with respect to the sub-string, \( H_{i,j}[k] \), beginning at phoneme index, \( k \), in the phonemic expansion of the path. The normalized score, \( D(i,j)_{s_i}(Q) = M(Q, H_{i,j}[k]) / M(Q, Q) \), is computed for all path offsets, all paths, and all index terms \( i \in I_Q \). This score is compared to a threshold and used for reporting the detection performance given in Section 5.

### 4. TASK DOMAIN

The task domain used for this study consists of audio recordings of course lectures obtained from the McGill Courses On-Line (COOL) repository [10]. There are a large number of stored lectures in the repository recorded in lecture room environments using a variety of microphones. Several of these recorded lectures were manually transcribed and used as development and evaluation data sets. The evaluation speech data consists of two lectures containing a total of 131 minutes of speech recorded through a lapel microphone from a single male speaker who speaks English as his third language. The test set transcriptions contain a total of 17914 words and the test utterances were segmented into 387 audio segments with an average segment length of 20 seconds.

Word lattices were generated off-line for each audio segment using an LVCSR system that was originally developed and evaluated under the AMI project [11] and was configured for this task as described in [1]. The feature analysis, acoustic models, and language models used for this particular system are described in [1]. The language model test set perplexity is 143 and, even with a vocabulary of 52,800 words, the OOV rate is a relatively high 11.2 percent. The word accuracy obtained for the lecture speech evaluation set was 56.5 percent.

To evaluate STD performance, a set of query terms were chosen from the words in the test set transcriptions based on their frequency of occurrence in the transcriptions. After removing functions words and low frequency words, a set of 177 of the most frequent words in the test set were chosen as query terms for the study in Section 5. The query terms consist of 143 IV words and 34 OOV words with respect to the LVCSR vocabulary. The length of the phonetic expansion of the query terms ranges from as few as 2 phonemes for “ear” to 13 phonemes for the term “neurotransmitter”. There are a total of 1930 occurrences of IV and OOV query terms in the test set transcriptions. This corresponds to an average of 8 occurrences per query term in the 131 minute test corpus.

Two measures are used here to evaluate performance. The first is a precision-recall measure and is used to describe the number of relevant and non-relevant audio segments that are retrieved for a given query term. An audio segment is considered relevant if it contains at least one occurrence of the query term. Precision is defined as the number of retrieved segments that contain the query term relative to the total number of retrieved segments. Recall is defined as the number of retrieved segments that contain a given query term relative to the total number of segments that include that term. These values are computed for each query term and averaged. This measure is used in Section 5 to evaluate the performance of the first pass search procedure described in Section 3.2.

The second performance measure evaluates the number of correct detections and false detections at the term level rather than the segment level. The average probability of detection and the number of false alarms per search term (keyword) per hour (fa/kw/hr) are computed for each search term and averaged. The probability of detection, \( P_d = N_d / N_t \), is defined as the number of correctly detected query terms, \( N_d \), normalized by the total number of actual query term occurrences in the test set, \( N_t \). A query term detection is considered to be “correct” if it occurs within a 2 second window surrounding the starting time of the actual occurrence of the term in the
test data. Otherwise it is labeled as a false alarm. This measure is used in Section 5 to look at the impact of the second pass verification approach on both the rate of false rejection and false detection of query terms.

5. Experimental Study

This section presents an experimental study for evaluating the effectiveness of the index construction and multiple pass search approaches presented in Section 3. First, the size of the index and computational complexity of index construction are addressed. Second, the degree to which index entries cover the occurrences of the query terms in the audio segments is investigated. Finally, results are given for spoken term detection performance obtained by the second pass detailed search procedure. Spoken term detection results are reported for the audio course lecture task domain described in Section 4.

5.1. Index Construction

The index is updated as new audio files are added to the repository. The issue of the memory and computational requirements associated with updating the index is discussed here. The size of the index was shown in Section 3.1 to be proportional to the number of audio segments and also proportional to the ASR vocabulary size. It should also be clear that as the boosting factor, $B$, in Equation 3 increases, the number of entries in the index will also increase.

The index size was estimated for an empirically chosen value of $B = 200$, which was shown to provide a reasonable trade-off between STD performance and index size. With this value of $B$ and a vocabulary size of $|V| = 52,800$ words, the index construction procedure described in Section 3.1 resulted in an average of approximately 42,000 $(p_{i,j},l_{i,j})$ pairs per hour of audio on the evaluation set. Depending on how much memory is required for storing the word string for a lattice path, this amounts to on the order of several Mbytes per hour of audio which is far less than the memory required to store the compressed audio files.

The algorithm described in Section 3.1 for creating the index entries given in Equation 1 simply identifies lattice paths that are likely to contain vocabulary words, $V_i$. This requires on the order of $|V|$ iterations of updating path likelihoods, $L'_{i,j}$, and rescoring lattices for each new audio segment. These operations are implemented in a finite state automata framework and applied off-line to new audio segments as they are added to the repository.

5.2. Index Coverage

Table 1 addresses the trade-off between index size and coverage of search terms in audio segments with respect to $B$ in Equation 3. The number of index entries, for a given $V_i$ in Equation 1 for the index construction method in Section 3.1 increases with $B$. Increasing the number of index entries results in a larger number of segments containing $V_i$ that can be retrieved during the first pass search. However, it also increases the number of retrieved segments that do not contain $V_i$. Table 1 corresponds to the case where the paths for only a single indexing term are retrieved for a given query. The table shows that as $B$ varies over a range from 0 to 300, the percentage of segments in the collection corresponding to actual occurrences of $Q$ that are retrieved increases from 52 to 85 percent. On the other hand, the percentage of retrieved segments corresponding to actual occurrences of $Q$ decreases from 94 to 53 percent. The precision and recall values in Table 1 were computed only for IV query terms. A value of $B = 200$ was used for the STD evaluation in Section 5.3.

<table>
<thead>
<tr>
<th>Boosting Factor</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>51.82</td>
<td>93.85</td>
</tr>
<tr>
<td>80</td>
<td>65.23</td>
<td>85.68</td>
</tr>
<tr>
<td>120</td>
<td>71.81</td>
<td>78.65</td>
</tr>
<tr>
<td>200</td>
<td>80.84</td>
<td>64.95</td>
</tr>
<tr>
<td>300</td>
<td>84.69</td>
<td>52.51</td>
</tr>
</tbody>
</table>

Table 1. Precision/recall performance of first past search in retrieving candidate segments for IV queries.

5.3. STD Performance

This section investigates the overall performance of the index based two-pass search procedure. The characteristics of the word-based index used for accessing candidate lattice paths in the first pass of the search procedure were described in Table 1. Figures 1 and 2 describe the spoken term detection performance for IV and OOV query terms respectively. Both figures plot the query term detection probability and the number of false alarms with respect to the number of indexing terms, $|I_Q|$, that are retained for each query term, $Q$, as described in Section 3.2. These are referred to as indexing entries (IE) in the figures. The second pass search for verifying the occurrence of the search term is performed only for the paths associated with the indexing terms retained in the first pass.

It is clear from Figure 1 that the detection probability increases approximately nine percent as the number of retained indexing entries increases from 1 to 10 indexing terms. Hence, retaining the paths associated with multiple indexing terms is important even when an exact match exists between the query term and one of the indexing terms $V_i \in V$. A best performance of approximately $P_d = 76$ percent at 10 fa/kw/hr is obtained for $IE = 10$.

Figure 2 shows similar behavior in STD performance for the OOV query terms. The baseline performance curve for this figure corresponds to performing exhaustive search of the phonetic expansion for the top ASR string candidate. The $P_d$ consistently improves with increasing IE in operating regions above 8 fa/kw/hr. At lower operating regions, searching the additional paths resulting from the higher number of index entries results in a slight increase in the number of false alarms at a given $P_d$. A best performance of approximately $P_d = 37$ percent at 10 fa/kw/hr is obtained for $IE = 20$. 

The algorithm described in Section 3.1 for creating the index entries given in Equation 1 simply identifies lattice paths that are likely to contain vocabulary words, $V_i$. This requires on the order of $|V|$ iterations of updating path likelihoods, $L'_{i,j}$, and rescoring lattices for each new audio segment. These operations are implemented in a finite state automata framework and applied off-line to new audio segments as they are added to the repository.
6. SUMMARY AND CONCLUSION

A novel indexing strategy was proposed for open vocabulary spoken term detection. The method is based on constructing a word-based index from ASR lattices and using an approximate subword-based algorithm for accessing index entries from subword expansions of query terms. A multiple pass search strategy that involves indentifying candidate lattice paths in the first pass and verifying the occurrence of these terms through a detailed phoneme based search in the second pass was presented. The experimental study demonstrated that the indexes can be efficiently updated as audio segments are added to the collection and the indexes themselves were shown to be of manageable size and scalable to large collections. STD performance was measured on the lecture speech task domain where baseline ASR word accuracy was found to be approximately 56 percent. A detection rate of $P_d = 76$ percent at 10 fa/kw/hr was obtained for IV query terms and $P_d = 37$ percent at 10 fa/kw/hr was obtained for OOV search terms.

7. ACKNOWLEDGMENTS

The authors would like to thank Vishwa Gupta at the Centre de Recherche Informatique de Montreal (CRIM) for important advice and suggestions on the techniques presented in this paper. We would also like to thank Martin Karafiat from Brno University of Technology (BUT) for supplying us with the AMI ASR system.

8. REFERENCES


