SPOKEN TERM DETECTION USING FAST PHONETIC DECODING

Roy Wallace, Robbie Vogt, Sridha Sridharan

Speech and Audio Research Laboratory,
Queensland University of Technology (QUT), Brisbane, Australia
royw@ieee.org, {r.vogt,s.sridharan}@qut.edu.au

ABSTRACT

While spoken term detection (STD) systems based on word indices provide good accuracy, there are several practical applications where it is infeasible or too costly to employ an LVCSR engine. An STD system is presented, which is designed to incorporate a fast phonetic decoding front-end and be robust to decoding errors whilst still allowing for rapid search speeds. This goal is achieved through monophone open-loop decoding coupled with fast hierarchical phone lattice search. Results demonstrate that an STD system that is designed with the constraint of a fast and simple phonetic decoding front-end requires a compromise to be made between search speed and search accuracy.

Index Terms— spoken term detection, speech recognition

1. INTRODUCTION

Our speech is being collected and stored in unprecedented volumes. The utilisation of the vast amounts of information held therein urgently requires the development of technologies that allow computers to make these collections accessible and useful for humans. There is demand from a vast range of industries including speech analytics, surveillance, consumer search and media monitoring.

Past efforts have focused on the task of Spoken Document Retrieval (SDR), in which systems are designed to retrieve spoken documents deemed to be relevant to a human user's query. After a series of National Institute of Standards and Technology (NIST) evaluations, this was declared a success [1], however the evaluations focused primarily on broadcast news, where relatively accurate large vocabulary continuous speech recognition (LVCSR) was possible, and queries were at least as long as a phrase or sentence written in natural language, which is unlikely to be the case in many practical applications. There is also some ambiguity in SDR, such as the subjective definitions of relevance and document length.

An alternative approach that has had a recent surge in interest is spoken term detection (STD), which involves the detection of all occurrences of a specified search term, usually a word or phrase, rapidly and accurately in large heterogeneous audio archives [2].

Generally, STD systems first pre-process the audio to create an index to allow for subsequent rapid searching. This index is compiled from an automatically generated transcription or lattice representation of the speech, in terms of words [3] or sub-word units, often phones [4, 5, 6]. A word-level index can provide for accurate term detection; however, the necessary LVCSR decoding is relatively slow, and new and rare terms cannot be easily detected at search time if they are out-of-vocabulary (OOV). A phonetic index, on the other hand, can be created relatively quickly and is inherently open-vocabulary. Whilst there have been recent efforts in fast LVCSR decoding for STD [6], it has been conceded that increases in LVCSR speed often increase transcription word error rate, which translates to degradation in STD accuracy [3].

Fusion of word and phonetic indices is an obvious extension and has been shown to consistently improve STD accuracy, even by simply using the word-level index for in-vocabulary terms and the phonetic index to support search for out-of-vocabulary terms [7, 8, 9]. This approach requires the use of an LVCSR engine during decoding; therefore, the phonetic index can be generated either from the expansion of the word transcription [6, 8], or alternatively using a separate phonetic decoder.

Unfortunately, there remain some applications where the use of an LVCSR engine during decoding is undesirable or simply infeasible, either due to the computational cost being too high, or the accuracy of the word-level decoding being insufficient in the particular domain of interest. There is thus demand for standalone phonetic indexing in applications where huge amounts of data are required to be indexed quickly, in languages and domains with insufficient data to train an accurate LVCSR system, and in applications where detection of OOV terms is of primary concern, for example in multilingual collections and for surveillance.

The design of an STD system is clearly very application-dependent, and various characteristics such as indexing speed, search speed, domain and expected kinds of search terms should be considered jointly. This is becoming apparent in the literature, for example in [10], where a phonetic indexing approach is presented that sacrifices detection accuracy for improved index size and search speed. The system in [10] still uses very slow indexing, which may present a problem in a practical deployment.

This paper presents follow-on work to [11, 12], and demonstrates the use of a phonetic system to provide fast, open-vocabulary search without the need for an LVCSR engine. This paper explores some of the trade-offs and trends influenced by STD system design, by introducing the use of fast and simple phonetic decoding (monophone models on conversational telephone speech), and demonstrating how the subsequent indexing and search phases can be adjusted to maintain detection accuracy and search speed.

2. SPOKEN TERM DETECTION SYSTEM

The system is based on the Dynamic Match Lattice Spotting technique, introduced in [11] and reported in [12] in the context of the NIST 2006 STD Evaluation [2]. Indexing is run once to produce a phonetic index that is independent of the search terms, followed by search for each term, using the index to detect term occurrences.

2.1. Indexing

The purpose of the speech indexing stage is to construct a database to provide fast and robust subsequent search. First, a phonetic speech

978-1-4244-2354-5/09/$25.00 ©2009 IEEE

Models	Phone recognition errors				Decoding speed
	PER	Sub	Ins	Del	
Tri-phone AM, phonotactic LM	42%	19%	4%	19%	3.3 x sRT
Mono-phone AM, open-loop	69%	35%	7%	27%	5.4 x fRT

Table 1. Phone error rates (PER) on eval data. Sub, Ins, and Del are contributions of substitution, insertion and deletion errors to PER.

recogniser is used to decode each speech segment, resulting in a lattice of multiple phone recognition hypotheses. A modified Viterbi traversal is then used to traverse these lattices and emit all phone sequences of a fixed length, N, that terminate at each node in the lattice. As in [12], $N = 11$ was chosen to provide a suitable compromise between index size and search efficiency. The resulting collection of phone sequences is then compiled into a sequence database (SDB), which is effectively a look-up table that returns the location of each occurrence of a particular indexing unit, in this case, a unique N-gram phone sequence - a common approach for storing both word and phonetic indices [3, 4, 6, 7].

Unlike [6, 10], rather than only indexing the 1-best phonetic transcription, paths throughout the lattice are stored in the index so that simpler and faster phonetic decoding can be used whilst still maintaining recall as much as possible. Lattice generation uses 3 tokens and a beam-width of 50, which was found to provide optimal STD accuracy whilst minimising decoding time and index size.

In contrast to [12], the goal of this paper is to use fast, simple phonetic decoding for STD. Therefore, for acoustic modeling, tied-state 16 mixture tri-phone HMM's have been replaced with 32 mixture mono-phone HMM's. Also, while [12] made use of 2-gram and 4-gram phonotactic language models, here an open phone loop is used. Comparative 1-best phone error rates are given in Table 1. This results in an approximately 18 times speed increase in decoding, from 3.4 times slower than real-time (sRT) to 5.4 times faster than real-time (fRT). Mono-phone decoding uses the HMM Toolkit (HTK) tool HVite, whereas tri-phone decoding uses HDecode.

The actual indexing speeds reported here are not particularly remarkable, due to the lack of optimisation of any kind of the decoder. Decoding has not been optimised for speed in any way other than the focus of this paper, that is, the use of simple acoustic models.

2.2. Search

Once the index has been constructed, the system can accept search terms in the form of a word or phrase. A pronunciation lexicon is used to convert the search term into a sequence of phones, referred to as the target sequence. Letter-to-sound rules can be used for out-of-vocabulary terms. Search then consists of retrieving phone sequences from the database that closely match the target sequence.

Search involves the calculation of the Minimum Edit Distance (MED) between each indexed phone sequence, $X = (x_i)_{i=1}^{U}$, and the target sequence, $Y = (y_i)_{i=1}^{V}$, using dynamic programming. Some simple optimisations are incorporated to minimise the number of calculations performed in the process [11]. This kind of "fuzzy" matching is a common approach to introduce some robustness to the high error rates of phonetic decoding [8, 9].

The MED is defined as the minimum possible sum of phone substitution, insertion and deletion costs that transform the sequence into the target sequence. The cost of an error is inversely related to the likelihood of it occurring. Substitution costs are defined as

$$C_s\left(x,y\right) = \begin{cases} -\log\left(p\left(R_y|E_x\right)\right) & x \neq y \\ 0 & x = y \end{cases}, \tag{1}$$

where $p\left(R_y|E_x\right)$ is the probability that phone y was actually uttered (according to the reference) given that phone x was emitted by the decoder. Insertion costs are defined in terms of the probability that there is no corresponding phone in the aligned reference given that phone x is emitted, i.e.

$$C_i\left(x\right) = -\log\left(p\left(R_*|E_x\right)\right). \tag{2}$$

Deletion costs are defined in terms of the probability that y is in the reference but no corresponding phone is emitted, i.e.

$$C_d\left(y\right) = -\log\left(p\left(R_y, E_*\right)\right). \tag{3}$$

Using Bayes Theorem, these statistics are computed from the insertion and deletion counts, recognition likelihoods, $p\left(E_x|R_y\right)$, phone prior probabilities, $p\left(R_y\right)$, and emission probabilities, $p\left(E_x\right)$, estimated from a recognition confusion matrix generated during development using the HTK tool, HResults. Whereas more complex methods of deriving costs are possible, such as using HMM divergence or confusion statistics trained from phonetic lattices, HResults was found in preliminary experiments to be sufficient and is also convenient for estimating insertion and deletion costs.

In [12], it was found that allowing for insertions and deletions did not greatly improve search accuracy, when used with tri-phone based phonetic decoding. It appears that this does not so much apply to the mono-phone decoding case. Trends found for this system using mono-phone decoding will be presented in section 4.

It should be clear that these costs introduce robustness to common decoding errors. It is interesting to note that, as the pronunciation lexicon is used to generate the reference phonetic transcript from which the confusion statistics are estimated, these costs also introduce robustness to any regular mismatches between the lexicon and the actual pronunciation of terms as they occur in the collection.

2.3. Hyper-sequence database

As described in [12], a mapping from phones to one of 5 phonetic classes (vowel, nasal, fricative, stop, liquid/glide) is used to generate a hyper-sequence database (HSDB), which is a constrained domain representation of the sequence database (SDB). Thus, each entry in the HSDB will map to a number of entries in the SDB, each of which is a sequence of phones that maps to the same sequence of phonetic classes. The resulting two-tier, hierarchical database structure can be used to reduce the search space and allow for rapid search.

Other forms of mapping using, e.g., data-driven clustering are possible, however the linguistic approach was found to perform comparatively well. This concept is similar to the metaphones of [9]. The use of the HSDB differs from the approach of [4, 5] in that the search space is first restricted to a set of phone sequences likely to have been generated by an utterance of the search term, instead of restricted to regions of speech likely to contain the search term.

Search in the HSDB using MED calculations can be implemented in exactly the same way as in the SDB, with phone class substitution, insertion and deletion costs trained from a corresponding phone class confusion matrix.

3. EVALUATION PROCEDURE

Accuracy is measured in terms of maximising the proportion of detected occurrences (detection rate), and minimising the number of false alarms. Due to the lack of discrete trials, false alarms are usually reported as a rate per hour. Detection rate can be plotted against false alarm rate to create a Receiver Operating Characteristic (ROC) plot. The popular and well established Figure of Merit (FOM) is a single figure between 0 and 1, defined as the average detection rate at integer values between 0 and 10 false alarms per search term per hour.

NIST's recently introduced metric, Actual Term-Weighted Value [2], is an alternative measure of the trade-off between detection and false alarm rate [10]. However, the ROC plot and FOM are more simply related to each other than the detection error trade-off curve and ATWV suggested by NIST. False alarm prevalence is also more accurately described as a rate, rather than a probability that is based on the subjective and synthetic concept of a non-target trial rate. And whilst the definition of the cost/value ratio for ATWV is somewhat analogous to the range of false alarm rates considered for FOM, the effect of the latter is more naturally represented in the graphical form of a ROC plot.

The same index is used in all experiments reported below. Phonetic decoding is performed at 5.4 times fRT. All indexing, including decoding and construction of the SDB and HSDB is 4.2 times fRT. Search speed is measured in hours of speech searched per CPU-second per search term (hrs/CPU-sec).

3.1. Evaluation data

Evaluation is performed on a 9 hour subset of the Fisher conversational telephone speech corpus, three times larger than that used in [2]. A separate 9 hour subset is used for development. A total of 1200 search terms are chosen randomly from a pool of words that occur at least once in the evaluation data, with 400 words selected for each of the lengths of 4 phones, 6 phones, and 8 phones (e.g. "nike", "baghdad", "olympics"). Whilst longer search terms are generally more successfully detected, especially by phonetic-based systems, these short terms were chosen so as to provide a lower bound on performance, which is necessary to prove that such a phonetic system can compete with LVCSR-based systems.

Whilst term selection is a critical step in ensuring a fair evaluation, it was decided here to avoid introducing bias by selecting terms randomly, rather than specially crafting a set of terms to match the kinds expected in a particular practical deployment. Compared to LVCSR-based systems, the impact of term selection on results should be diminished, as no word-level language model is used in indexing or search. Specifically, results for such systems may be affected by the proportion of search terms that are in-vocabulary or out-of-vocabulary. Even if they also incorporate a back-off to a sub-word index, the sequences therein are likely to be biased towards those that occur within the in-vocabulary terms. In a system such as the one described here, there is, more or less, simply no vocabulary.

4. EXPERIMENTAL RESULTS

Table 2 shows the trends in performance for various search configurations, and the apparent trade-offs between search accuracy and speed, given the use of fast phonetic decoding during indexing.

Initially, the HSDB is not used, meaning that search is performed on the entire SDB. One important parameter is the kind of errors that are accommodated in MED scoring, that is, substitution, insertion and/or deletion errors. If only substitution errors are allowed, as in [12], an FOM of 0.201, 0.309, and 0.301 for 4, 6 and 8-phone terms is achieved, at a search speed of 11 hrs/CPU-sec.

Allowed errors		Figure of Merit			Search speed
HSDB	SDB	4-phn	6-phn	8-phn	(hrs/CPU-sec)
-	S	0.201	0.309	0.301	11
-	S, I	0.198	0.311	0.307	1
-	S, D	0.201	0.316	0.383	2
-	S, I, D	0.198	0.318	0.400	1
None	S	0.201	0.242	0.117	164
S	S	0.201	0.313	0.301	25
S, I	S, I	0.204	0.317	0.297	12
S, D	S, D	0.201	0.317	0.393	10
S, I, D	S, I, D	0.204	0.323	0.398	9

Table 2. STD accuracy (FOM), where various combinations of substitution (S), insertion (I), and deletion (D) errors are allowed for with associated costs, in the HSDB and SDB. An entry of "-" under HSDB indicates that the entire SDB is searched.

The constraint of only allowing for substitutions greatly reduces the complexity of MED calculation.

If insertion or deletion errors are allowed, the FOM is improved for 6 and 8-phone terms, at the cost of reducing search speed by around a factor of 10. Allowing for deletion errors is particularly important for 8-phone terms, providing a 27% relative improvement.

The benefits of allowing for substitution, insertion and deletion errors are complementary for 6 and 8-phone terms, leading to 3% and 33% relative improvements. However, this comes at the cost of much slower search speed. In contrast, using the slower decoding of [12], allowing insertion and deletion errors similarly slows down search speed but does not give as substantial relative gains in FOM (from 0.249, 0.515 and 0.575, relative improvement is less than 1%, 1% and 5% for 4, 6-phone and 8-phone terms respectively).

4.1. Use of hyper-sequence database

To increase search speed whilst maintaining accuracy, the HSDB is introduced as a preliminary search phase to narrow the search space to a subset of the SDB, reducing the number of necessary MED calculations.

The simplest and fastest method of HSDB search, as used in [12], is to first determine the hyper-sequence (i.e. sequence of phone classes) representing the search term, and select from the SDB only those sequences that map to the same hyper-sequence, as candidates for MED-based search. This effectively only allows for matching sequences where phones may have been substituted with a phone in the same class.

This results in very fast search, at 164 hrs/CPU-sec; however, accuracy is sacrificed by 22% and 61% for 6 and 8-phone terms. With the slower decoding of [12], an even larger search speed increase is achieved (from 14 to 422 hrs/CPU-sec), but accuracy is not as severely affected (reduced by 5% and 12% for 6 and 8-phone terms). Here, unusually, results are much worse for longer terms. Search for 4-phone terms is not affected, suggesting that phone deletions become more of a problem for longer search terms.

To make a compromise between the speed benefit of using the HSDB and the accuracy of searching the entire SDB, rather than selecting only a single hyper-sequence, it is possible to select a subset of hyper-sequences. An MED score is calculated between each hyper-sequence in the HSDB and the target hyper-sequence. This is still used to limit the search space of the SDB, but to a lesser extent, by tuning an MED threshold, T.

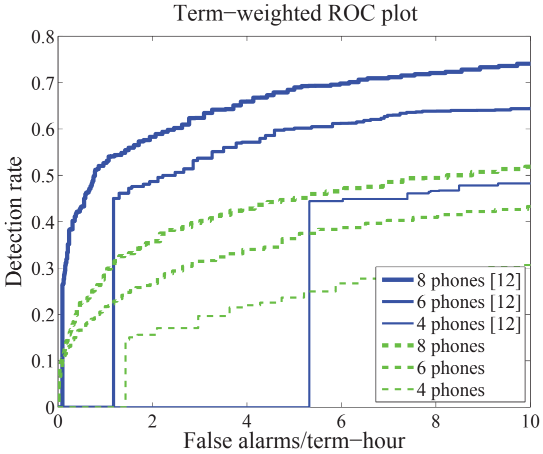

Fig. 1. STD accuracy (ROC plot) where substitutions, insertions and deletions are allowed in HSDB and SDB, using tri-phone decoding as in [12], or faster mono-phone decoding. Detection rate is averaged across all terms of a particular phone length (i.e. term-weighted).

As shown in Table 2, this approach can be used to entirely maintain search accuracy whilst greatly increasing search speed. If substitution errors alone are considered, search speed is increased from 11 to 25 hrs/CPU-sec. If substitution, insertion and deletion errors are allowed, search speed is increased from 1 to 9 hrs/CPU-sec. With the slower decoding of [12], this use of the HSDB increases search speed from 2 from 32 hrs/CPU-sec, likewise with no loss in FOM. For 4 and 6-phone terms, accuracy is even slightly improved, suggesting that the use of the HSDB eliminates a number of false alarms that would otherwise have scored reasonably well in the SDB search.

4.2. Summary

The ROC plot displayed in Figure 1 shows that with fast decoding, over 50% of 8-phone terms are detected at a rate of 10 false alarms per hour. The plot shows that in some situations, the first possible operating point is at a false alarm rate much greater than 0. This is the point where only exactly matching sequences are output. Because they all have an MED of zero, they cannot be separated from one another. This affects the FOM, and is a problem mainly for very short terms that can be alleviated by increasing the resolution of match scores or reducing the false alarm rate, e.g. by incorporating confidence scoring using acoustic scores (as in [12]).

The trade-off between search speed and STD accuracy is illustrated by Figure 2. The adjacent points where accuracy is maintained whilst search speed is increased show the loss-less nature of the incorporation of the HSDB search, when either substitution errors, or all error types, are allowed for.

5. CONCLUSIONS

A spoken term detection system is presented, using a phonetic search technique that works with a fast phonetic decoding front-end. With 4.2 times fRT indexing, 8-phone terms are detected with an FOM of 0.398, using an HSDB to increase search speed almost by an order of magnitude to 9 hrs/CPU-sec.

Decisions regarding the trade-offs between indexing speed, search speed and detection accuracy have been demonstrated to influence STD system design. The system is vocabulary-independent and relatively flexible for porting to other domains and languages.

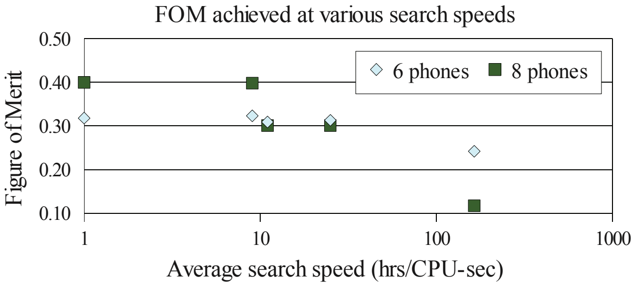

Fig. 2. Figure of Merit (FOM) achieved for a selection of configurations from Table 2.

While speech recognition accuracies continue to improve, it will be interesting to see how spoken term detection systems as a whole can be improved through design according to the constraints of the task and application.

6. REFERENCES

[1] J.S. Garofolo, C.G.P. Auzane, and E.M. Voorhees, “The TREC spoken document retrieval track: A success story,” *Proc. TREC-8*, 2000.

[2] National Institute of Standards and Technology, “The Spoken Term Detection (STD) 2006 evaluation plan,” September 2006.

[3] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, “The SRI/OGI 2006 spoken term detection system,” in *Proc. Interspeech*, 2007, pp. 2393–2396.

[4] S. Dharanipragada and S. Roukos, “A multistage algorithm for spotting new words in speech,” *IEEE Trans. Speech Audio Process.*, vol. 10, no. 8, pp. 542–550, 2002.

[5] P. Yu, K. Chen, C. Ma, and F. Seide, “Vocabulary-independent indexing of spontaneous speech,” *IEEE Trans. Speech Audio Process.*, vol. 13, no. 5, pp. 635–643, 2005.

[6] U. Chaudhari, Hong-Kwang Jeff Kuo, and B. Kingsbury, “Discriminative graph training for ultra-fast low-footprint speech indexing,” in *Proc. Interspeech*, 2008.

[7] I. Szoke et al., “BUT system for NIST STD 2006 - english,” in *Proc. NIST Spoken Term Detection Evaluation workshop*, 2006.

[8] C. Dubois and D. Charlet, “Using textual information from LVCSR transcripts for phonetic-based spoken term detection,” in *Proc. ICASSP*, 2008, pp. 4961–4964.

[9] J. Mamou, Y. Mass, B. Ramabhadran, and B. Sznajder, “Combination of multiple speech transcription methods for vocabulary independent search,” in *SIGIR ’08*, 2008.

[10] J. Pinto, I. Szoke, S.R.M. Prasanna, and H. Hermansky, “Fast approximate spoken term detection from sequence of phonemes,” in *SIGIR ’08*, July 2008, pp. 28–33.

[11] K. Thambiratnam and S. Sridharan, “Rapid yet accurate speech indexing using Dynamic Match Lattice Spotting,” *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 1, 2007.

[12] R. Wallace, R. Vogt, and S. Sridharan, “A phonetic search approach to the 2006 NIST Spoken Term Detection evaluation,” in *Proc. Interspeech*, August 2007, pp. 2385–2388.