EFFICIENT SEARCH USING POSTERIOR PHONE PROBABILITY ESTIMATES

Steve Renals

Department of Computer Science
University of Sheffield
Sheffield S1 4DP, England
S.Renals@dcs.shef.ac.uk

Mike Hochberg

Department of Engineering
University of Cambridge
Cambridge CB2 1PZ, England
mmh@eng.cam.ac.uk

ABSTRACT

In this paper we present a novel, efficient search strategy for large vocabulary continuous speech recognition (LVCSR). The search algorithm, based on stack decoding, uses posterior phone probability estimates to substantially increase its efficiency with minimal effect on accuracy. In particular, the search space is dramatically reduced by *phone deactivation pruning* where phones with a small local posterior probability are deactivated. This approach is particularly well-suited to hybrid connectionist/hidden Markov model systems because posterior phone probabilities are directly computed by the acoustic model. On large vocabulary tasks, using a trigram language model, this increased the search speed by an order of magnitude, with 2% or less relative search error. Results from a hybrid system are presented using the Wall Street Journal LVCSR database for a 20,000 word task using a backed-off trigram language model. For this task, our single-pass decoder took around 15× realtime on an HP735 workstation. At the cost of 7% relative search error, decoding time can be speeded up to approximately realtime.

1. INTRODUCTION

The development of efficient search procedures is becoming an increasingly important area of large vocabulary continuous speech recognition (LVCSR). The search problem is to locate the most probable string of words for a spoken utterance given the acoustic signal and sentence models. Evaluation of the search space, which is large due to the vocabulary size, is made more complex when long-span language models (LMs) are used.

This paper describes an efficient search algorithm which has been incorporated in ABBOT, a hybrid connectionist/hidden Markov model (HMM) LVCSR system [1]. Hybrid connectionist/HMM systems, such as ABBOT, use connectionist networks to compute direct local estimates of posterior phone probabilities [2, 3]. These local posterior probabilities are converted to scaled likelihoods and integrated into the HMM framework as (scaled) estimates of HMM output likelihoods. At recognition time these scaled likelihoods are used in the decoding, rather than directly using the posterior probability estimates. ABBOT uses a recurrent network [4] to estimate the phone output probabilities of a context-independent HMM system.

This acoustic modelling approach is somewhat different to the context-dependent mixture model approach used in most other systems and imposes different constraints on the search. In particular the following differences have proved to be important in the design of an efficient decoder:

- Direct estimation of posterior probabilities, $P(\text{phone} \mid \text{data})$, by the network, rather than likelihoods, $p(\text{data} \mid \text{phone})$;
- Context-independent acoustic modelling leads to a small set of basic HMMs (typically 40–80), rather than several thousand context-dependent models;
- Network probability estimation enables the computation of all phone probability estimates at each frame without much additional computational cost.

In this paper we describe a new search algorithm that we have developed. This algorithm is based on the ideas of stack decoding [5, 6]. It operates in a partially time-asynchronous single pass and was designed for use with long-span language models. A novel aspect of this algorithm is a new pruning strategy, *phone deactivation pruning*. This approach makes direct use of the local posterior probability estimates computed by a connectionist network and may be used in conjunction with existing likelihood-based approaches.

This search algorithm has been implemented in a decoder referred to as NOWAY. Results are presented for the ARPA North American Business News (NAB) LVCSR database using a 20,000 word vocabulary and a backed-off trigram language model. The performance of the search algorithm was evaluated in terms of word error rate and computational resource requirement (CPU time and memory usage). We have been particularly concerned with the effect of varying the degree of posterior- and likelihood-based pruning and the results of an extensive set of experiments are reported.

2. SEARCH ALGORITHM

2.1. Basic Organization

The search algorithm described here is partially time-asynchronous and is based on the ideas of stack decoding [7, 8, 9, 5, 6]. The Viterbi criterion is used—*i.e.*, the full likelihood is not computed—so the algorithm may be regarded as a reordered time-synchronous Viterbi search. For simplicity of presentation, we consider decoding a single utterance of length T.

The basic data structure of the search algorithm is a priority queue, or *stack*. A stack supports the usual operations of a priority queue (*e.g.*, `pop` and `insert`), along with operations needed for the decoding task, such as `merge` and `replace`. The elements of the stack are *hypotheses*; a hypothesis h contains a proposed decoding W_h up to a given reference time t_h with a log likelihood L_h. W_h is comprised of a word sequence $\{w_h(0), w_h(1), \ldots\}$.

The *lexicon* is a list of pronunciations for each word in the vocabulary. For efficiency, it is stored as a tree; this reduces the number of constituent phone models required by a factor of 3 or 4, and allows computation to be shared shared between words with a similar head. Multiple pronunciations are stored as individual lexical items. A *node* in the tree corresponds to a phone in a particular set of pronunciations. It contains the topology and probabilities of the relevant phone HMM. The root node of the tree corresponds to a pause model—a single state silence model which may be skipped—to allow for optional inter-word pauses.

We assume a long-span language model. The basic operation of the LM is to provide the probability of an extension word w to a hypothesis h, *i.e.*, $P(w \mid w_h(0), w_h(1), \ldots, w_h(n))$. If a trigram language model is used, this probability is approximated by $p(w \mid w_h(n-1), w_h(n))$. Note that if a long-span LM is used, the most probable hypothesis for a complete utterance is not necessarily the most probable partial hypothesis at a time $t < T$.

2.2. Scoring Hypotheses

A fundamental decision that must be made in the stack decoding algorithm, is which hypothesis in the stack should be popped and extended. The criterion optimized in A^* search [10] may be stated as:

$$f_h(t_h) = L_h + b^*(t_h)$$

where $f_h(t)$ is an estimate of the log likelihood of an utterance with hypothesis h (with reference time t_h and log likelihood L_h), and $b^*(t_h)$ is an estimate of the log likelihood of the best extension of any hypothesis to the end of the utterance.

Direct computation of $b^*(t_h)$ implies looking ahead to the end of the data. This may be done using either a fast-match or multi-pass decoding (e.g. [11]). However, lookahead may be avoided if an approximation to the A^* criterion is used [5, 6] in which the difference between L_h and a least upper bound on the log likelihood of any hypothesis at that time is computed, *i.e.*,

$$f'_h(t_h) = L_h - \text{lub}L(t_h) ,$$

where $\text{lub}L(t_h)$ is the least upper bound on L_h at time t_h. In practice $\text{lub}L(t_h)$ is an estimate that may be updated as new hypotheses are examined. Using this approximation, hypotheses need only be compared with other hypotheses with the same reference time. This implies using a set of stacks: one for each time frame of the utterance to be decoded. This approach has been used successfully by Bahl and Jelinek [5] and Paul [6].

In our work, an initial estimate of $\text{lub}L(t_h)$ is generated from the network outputs. The n most probable phone posteriors (not including the most probable) are averaged and converted to a scaled likelihood by dividing by a uniform prior. This is similar to a garbage model technique used in wordspotting [12]. This estimate of $\text{lub}L(t_h)$ is then updated whenever a particular hypothesis extension has a higher likelihood at t_h.

2.3. Description of Algorithm

The basic algorithm is:

1. Set $t = 0$; $\text{lub}L(\tau) = -\infty$, $0 \leq \tau < T$; Initial null hypothesis: $t_h = 0$; $L_h = 0$ and $W_h =$ `NULL`.
2. Push initial hypothesis onto *stack*(0).
3. If (end-of-utterance) output top of *stack*(t) and exit.
4. Else process *stack*(t):
   - Pop all hypotheses into active hypothesis list, *hlist*.
   - If *hlist* is not empty expand hypotheses in parallel:
     - Activate root node of lexical tree
     - Propagate hypotheses forward time-synchronously and activate new nodes
     - Prune active nodes according to likelihood-based and posterior-based pruning criteria
     - Update $\text{lub}L(t)$ if required
     - At word-end nodes within envelope, extend hypotheses by one word, incorporate exact LM score, push hypotheses onto relevant stack.
     - Continue if any nodes are active
5. $t \leftarrow t + 1$; goto 3

The algorithm may be considered time-synchronous in that stacks of hypotheses are processed in sequential order. It may be considered time-asynchronous because when a hypothesis is popped from the stack and expanded, an arbitrary number of hypothesis extensions may be generated for various times in the future. This approach has also been termed "start-synchronous".

The bulk of the work is done when propagating the active hypotheses forward in parallel. This processing takes advantage of the tree-structured lexicon. The set of hypotheses may be propagated through the same tree and share acoustic information with their scores differing only in LM information and start scores. The tree is searched in a time-synchronous, breadth-first manner, although there is no *a priori* reason for preferring this to a depth-first search.

3. PRUNING

The search space of an LVCSR system using a long span LM is large and complex. If the search is to be manageable, then approximations must be made to reduce the effective size of the search space. The placing of such restrictions may be regarded as pruning certain phones, words or hypotheses from the search space without computing the complete probabilities for the relevant hypotheses.

3.1. Likelihood-based Pruning

In the usual maximum likelihood HMM systems, the search space is evaluated by computing likelihood estimates of the acoustic data having been generated by a particular utterance model. Pruning strategies are generally likelihood-based and involve the specification of a *likelihood envelope* Δ around the likelihood L of the most probable partial hypothesis at time t. Only hypotheses whose likelihood falls within the envelope (i.e those hypotheses with a likelihood $L' \geq L - \Delta$) remain in consideration. The other hypotheses are pruned. In the simplest form of likelihood-based pruning, the envelope is defined statically at all times. An adaptive likelihood envelope may also be defined by imposing a fixed upper limit on the number of hypotheses to be extended and evaluated at any given time. Efficient pruning can be enhanced by structuring the lexicon as a tree, in which the nodes correspond to phone models, and every path from the root to a leaf (or a node at a word end) corresponds to a pronunciation of a word in the dictionary [13].

3.2. Posterior-based Pruning

The phone posterior probabilities estimated by the network may be regarded as a local estimate of the presence of a phone at a particular time frame. If the posterior probability estimate of a phone given a frame of acoustic data is below a threshold, then all words containing that phone at that time frame may be pruned. This can be efficiently implemented within a tree structured lexicon. We refer to this process as *phone deactivation pruning*. The posterior probability threshold used to make the pruning decision may be empirically determined using a development set and is constant for all phones. The effect of varying the threshold on both recognition accuracy and CPU time is reported in section 5.

Phone deactivation pruning takes advantage of the fact that our basic acoustic component estimates posterior probabilities rather than likelihoods. Posteriors may be regarded as discriminative probabilities and do not give an estimate of P(data). Direct estimation of posteriors saves summing over baseform HMMs, which would be required to carry out an equivalent approach in a likelihood-based system. The channel-bank-based approach of Gopalakrishnan et al. [14] does attempt to use likelihoods to carry out a similar operation to phone deactivation pruning. However, this approach is somewhat more complex and requires phone-dependent thresholds. When their approach was used in a fast match, a factor of two speedup was achieved with a 5–10% increase in search error.

A second form of posterior-based pruning was also employed in NOWAY. This used local phone posterior estimates to spot leading and trailing silence. If the posterior estimate for silence in the beginning frames of an utterance is always above a threshold (*e.g.*, $P(\text{silence} \mid \text{data}) > 0.97$) then only the pause model (root node) is activated (and similarly for trailing silence).

4. LANGUAGE MODEL

The use of a language model is essential to constrain the search space in large vocabulary recognition. However there is a tradeoff between accessing the required language model probabilities, and the efficiency gain obtained in the search by their application.

The language models that we have used have been the standard back-off bigrams and trigrams. Since memory is at a premium in large-vocabulary search, the sparse arrays of trigrams and bigrams are stored compactly in computer memory and retrieval requires some computation. To aid efficient retrieval, we have used an incremental caching scheme in which LM probabilities are cached as they are used.

In a tree-based lexicon, the correct way to take advantage of the LM to reduce the search is by computing the maximum LM probability for each node. This involves taking a maximum over the LM probabilities of all words that use that node in their pronunciation given the hypotheses that they are extending. This involves a significant amount of computation, particularly in nodes close to the root of the tree which are part of the pronunciations of many words. In these cases we use an approximated upper bound, namely the maximum bigram given a context, $\max_k P(k \mid j)$, where j and k are the previous and current word labels respectively. This set of default bigrams is computed in advance and stored in a table. Experiments have indicated that using this approximation is more efficient at all word-internal stages of the search and the exact LM probabilities are used only at word ends. Incremental LM caching is still used at word ends, giving a 50% speedup. In this case, all hypotheses are propagated in parallel and only individually evaluated at word ends. Experiments in which individual hypotheses may be separately pruned at each node have been carried out, but do not show any efficiency improvements.

20K Trigram, Trained on SI-84					
Pruning Parameters		si_dt_s5		Nov '92	
Envelope	Stack Size	Time	Error	Time	Error
10	63	17.1	12.1	21.6	12.5
10	31	16.1	12.1	15.7	12.6
10	15	11.9	12.1	11.3	12.6
10	7	9.5	12.0	8.9	12.9
8	63	6.1	12.2	5.3	12.8
8	31	5.4	12.2	4.9	12.8
8	15	4.8	12.2	4.4	12.8
8	7	4.2	12.0	3.8	13.1

Table 1: Decoding performance on 20K trigram task. si_dt_s5 is closed vocabulary (actual size 5k) and Nov '92 is an open vocabulary (actual size 64K), with 2% OOV. Both sets are non-verbalized punctuation. The posterior probability threshold for phone deactivation pruning was set to 0.000075. The envelope is $\log_{10}$ scale, time is scaled in "realtime" units (on an HP735), error is the % word error. The actual size of the search process on an HP735 ranged from 102–129Mb.

5. EXPERIMENTS

We have experimented with this search procedure, as implemented in the NOWAY decoder, using the ARPA NAB CSR database. Our experiments have used a 20,000 word vocabulary and the 1993 standard backed-off trigram language model provided by MIT Lincoln Laboratories. The acoustic model used was a combination of recurrent networks using 78 phone classes plus silence trained on the WSJ0 short term speaker data (SI-84) [15][1]. We used the 20K pronunciation dictionary developed by Dragon Systems.

The aim of these experiments was to evaluate the search algorithm in terms of word error and computational requirement (CPU time and memory usage). In particular, we investigated the behaviour of the search algorithm relative to three pruning parameters: envelope, maximum stack size and phone posterior threshold.

The experiments were carried out using two data sets. The first, labelled si_dt_s5[2], contained 216 utterances from 10 speakers. This was a 5,000 word closed vocabulary set (the sentences being filtered from a larger open vocabulary set). The second set, labelled Nov '92[3], contained 333 utterances from 8 speakers. This used a 64,000 word vocabulary: about 1.9% of the words were out of vocabulary with respect to the 20,000 word dictionary. In the experiments here, the 20,000 word dictionary and standard trigram language model were used for both sets.

Comparative results using the NOWAY decoder are presented in tables 1 and 2. Table 1 shows how the performance varies relative to the likelihood-based pruning parameters, envelope and

[1] Note that most published results on this task have used the considerably larger WSJ1 (SI-284) acoustic data training set.

[2] This was the ARPA 1993 spoke 5 development set, using a Sennheiser microphone.

[3] This was the ARPA 1992 20K open NVP evaluation set.

20K Trigram, Trained on SI-84					
Pruning Parameters		si_dt_s5		Nov '92	
Envelope	Threshold	Time	Error	Time	Error
10	0.0	165.3	12.2	175.1	12.4
10	0.000075	16.1	12.1	15.7	12.6
10	0.0005	4.3	12.2	3.9	12.9
10	0.003	1.4	14.3	1.3	14.9
8	0.0	46.8	12.5	50.4	12.6
8	0.000075	5.4	12.2	4.9	12.8
8	0.0005	1.7	12.6	1.5	13.6
8	0.003	0.6	15.0	0.6	15.8

Table 2: Decoding performance with respect to varying phone deactivation pruning threshold. The maximum stack size was set to be 31. In cases when the posterior-based pruning threshold was greater than 0.0, posterior-based pruning of leading silence was also employed.

stack size. Table 2 illustrates the decoding performance relative to the phone deactivation pruning threshold. We note that applying posterior-based pruning with a threshold of 0.000075 gives around an order of magnitude improvement in the decoding speed with an increased relative search error of less than 2%.

The best parameter setting for realtime decoding is not shown in the tables above. However using a posterior threshold of 0.0005, an envelope of 8 and a stack size of 7 results in realtime performance (on an HP735) with a relative search error of around 7%.

We have recently applied NOWAY to a task using a vocabulary of 65,000 words. When applied to the 1994 ARPA CSR evaluation set (H1-P0) a decoding speed of 20 $\times$ realtime was obtained with 2% relative search error (13.0% word error) using a posterior threshold of 0.000075, an envelope of 9 and a stack size of 15 [1].

6. SUMMARY

We have described an approach to large vocabulary search for a hybrid connectionist/HMM system. Substantial improvements in efficiency, with little or no search error, have been achieved using features unique to the hybrid approach: local phone posterior probability estimates and a small set of context-independent HMMs. When applied to a task using a 20,000 word vocabulary and a trigram language model, the method of phone deactivation pruning offered an order of magnitude speedup with minimal effect on search accuracy and realtime decoding at a cost of 7% relative search error.

7. ACKNOWLEDGMENTS

S. Renals was supported by an EPSRC Postdoctoral Fellowship, while at Cambridge University. This work was partially supported by ESPRIT BRA 6487, WERNICKE.

8. REFERENCES

[1] M. M. Hochberg, S. J. Renals, A. J. Robinson, and G. D. Cook. Recent improvements to the ABBOT large vocabulary CSR system. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, Detroit, 1995.

[2] H. Bourlard and N. Morgan. *Connectionist Speech Recognition—A Hybrid Approach*. Kluwer Academic, 1994.

[3] S. Renals, N. Morgan, H. Bourlard, M. Cohen, and H. Franco. Connectionist probability estimators in HMM speech recognition. *IEEE Transactions on Speech and Audio Processing*, 2:161–175, 1994.

[4] A. J. Robinson. The application of recurrent nets to phone probability estimation. *IEEE Transactions on Neural Networks*, 5:298–305, 1994.

[5] L. R. Bahl and F. Jelinek. Apparatus and method for determining a likely word sequence from labels generated by an acoustic processor. US Patent 4,748,670, May 1988.

[6] D. B. Paul. An efficient A* stack decoder algorithm for continuous speech recognition with a stochastic language model. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 25–28, San Francisco, 1992.

[7] F. Jelinek. Fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development*, 13:675–685, 1969.

[8] L. R. Bahl and F. Jelinek. Decoding for channels with insertions, deletions and substitutions with applications to speech recognition. *IEEE Transactions on Information Theory*, 21:404–411, 1975.

[9] L. R. Bahl, F. Jelinek, and R. L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5:179–190, 1983.

[10] N. J. Nilsson. *Problem Solving Methods of Artificial Intelligence*. McGraw-Hill, New York, 1971.

[11] P. Kenny, R. Hollan, V. N. Gupta, M. Lennig, P. Mermelstein, and D. O'Shaughnessy. A*-admissible heuristics for rapid lexical access. *IEEE Transactions on Speech and Audio Processing*, 1:59–58, 1993.

[12] H. Bourlard, B. D'hoore, and J.-M. Boite. Optimizing recognition and rejection performance in wordspotting systems. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 373–376, Adelaide, 1994.

[13] R. Haeb-Umbach and H. Ney. Improvements in beam search for 10 000-word continuous-speech recognition. *IEEE Transactions on Speech and Audio Processing*, 2:353–356, 1994.

[14] P. S. Gopalakrishnan, D. Nahamoo, M. Padmanabhan, and M. A. Picheny. A channel-bank-based phone detection strategy. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 161–164, Adelaide, 1994.

[15] M. M. Hochberg, S. J. Renals, A. J. Robinson, and D. J. Kershaw. Large vocabulary continuous speech recognition using a hybrid connectionist–HMM system. In *Proceedings International Conference on Spoken Language Processing*, pages 1499–1502, Yokohama, Japan, 1994.