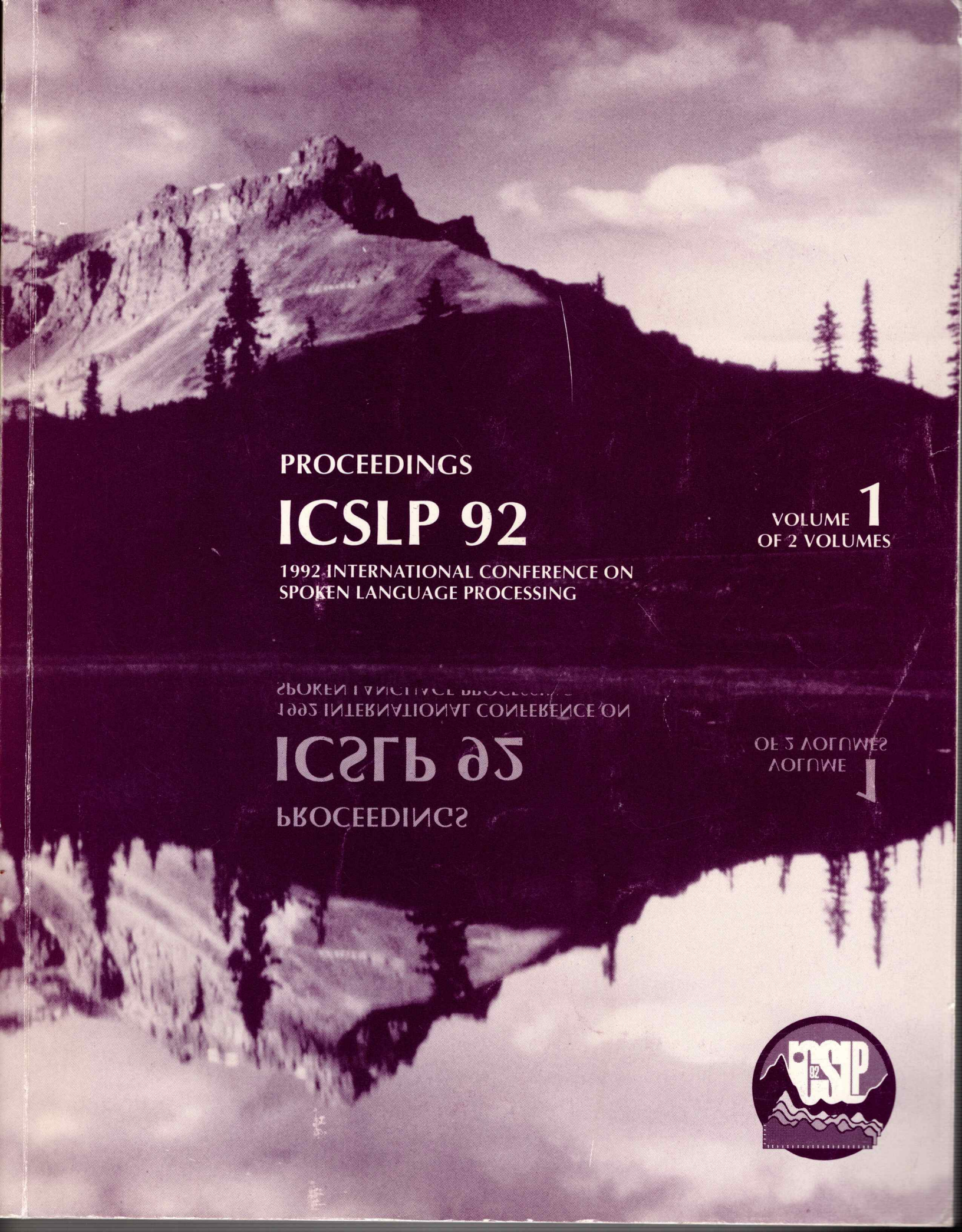

PROCEEDINGS

ICSLP 92

1992 INTERNATIONAL CONFERENCE ON
SPOKEN LANGUAGE PROCESSING

VOLUME 1
OF 2 VOLUMES

M. Lennig

ICSLP 92 PROCEEDINGS

International Conference on Spoken Language Processing

October 12–16, 1992
International Conference Centre
Banff, Alberta, Canada

Sponsored by

The University of Alberta

in conjunction with

Acoustical Society of America

Acoustical Society of Japan

Canadian Acoustical Association

Canadian Linguistic Association

European Speech Communication Association

IEEE — Signal Processing Society

International Phonetic Association, and

The Linguistic Society of America

VOLUME 1 OF 2

EXPERIMENTS IN CONTINUOUS SPEECH RECOGNITION WITH A 60,000 WORD VOCABULARY[1]

P. Kenny, R. Hollan[2], G. Boulianne, H. Garudadri, Y.-M. Cheng, M. Lennig and D. O'Shaughnessy

INRS-Télécommunications
16, place du Commerce
Verdun (Ile-des-Soeurs), Québec
Canada H3E 1H6

Abstract

We present a new search algorithm for very large vocabulary continuous speech recognition. Continuous speech recognition with this algorithm is only about 10 times more computationally expensive than isolated word recognition. We report preliminary recognition results obtained by testing our recognizer on "books on tape" using a 60,000 word dictionary.

1. INTRODUCTION

In this paper we will report our initial efforts to extend our earlier work on very large vocabulary isolated word recognition [16, 17, 19] to continuous speech tasks. We are aiming to perform speaker-dependent continuous speech recognition using a trigram language model and a vocabulary of 50,000–100,000 words.

Although the problem of very large vocabulary isolated word recognition has largely been solved [1, 19], no experiments have yet been conducted in continuous speech recognition with comparably large vocabularies. Probably the principal reason for this is that the search problem is so formidable. The best known approach to the search problem uses the word as the fundamental search unit and a stack decoding algorithm [11, 20] (also known as an A* search [14]). The effectiveness of this approach depends on having a good fast match strategy to identify candidate words whenever a word boundary is hypothesized. Many different fast match algorithms have been proposed [3, 4, 5, 15] but they have yet to be shown to perform satisfactorily on continuous speech tasks having very large vocabularies.

An alternative approach developped by Phillips [6] (on a 10,000 word vocabulary in German) uses the phoneme as the fundamental search unit and consists of a Viterbi search of the hidden Markov model obtained by combining phoneme HMMs with a Markovian language model (such as a trigram model). Aggressive pruning is necessary since the search space is very large. (For instance, if a trigram language model is used then one copy of the lexical tree is needed for every possible bigram.) The phoneme inventory is sufficently small that exact matches can be calculated whenever they are needed. However, it remains to be seen whether the phoneme unit is capable of attaining respectable accuracies on very large-scale recognition tasks.

A new type of bi-directional search strategy has emerged recently. The basic idea is to guide the search by means of a heuristic obtained by first carrying out an inexpensive search in the reverse time direction (subject to relatively weak linguistic and/or lexical constraints). This type of approach appears to have been discovered independently by several groups and has been shown to work effectively on a variety of applications [8, 9, 10, 16]. In [16] we presented a very efficient algorithm for very large vocabulary isolated word recognition using this paradigm and the phoneme as the fundamental search unit. In this paper we will describe our experience in extending this algorithm to continuous speech.

This isolated word recognition algorithm is an A* algorithm which uses a heuristic obtained by searching a phonetic graph which imposes triphone phonotactic constraints on phoneme strings and which enables us to identify the endpoint of the third-to-last phoneme in each partial recognition hypothesis with a high degree of accuracy, thereby substantially reducing the size of the search space. The acoustic matches of every segment of data with each of the phoneme models (the 'point scores') are precomputed before carrying out the A* search; this approach enables segment-level features such as phoneme durations to be modelled in an optimal way [16, 18, 7].

When applied to speaker-dependent isolated word recognition with a 60,000 word vocabulary, most of the computation is taken up by the pre-processing (the calculation of the point scores and the Viterbi search) and the A* search itself accounts for only about 1% of the total. In extending the algorithm to continuous speech recognition under the same conditions, we have found that the amount of pre-processing per unit time remains essentially the same, but the amount of computation needed for the A* search increases by three orders of magnitude. Hence, *the total computational demands of the algorithm only increase by a factor of about 10.*

The experiments reported here have been conducted using context-independent phoneme models and allophone models defined by contexts which do not extend across word boundaries. Although the search algorithm can be extended to accommodate cross-word allophone models fairly easily, our experience leads us to believe that if the number of allophone contexts is large (say several thousand) then considerations of efficiency will require a two-pass approach in which detailed allophone models are not used in searching but only in rescoring hypotheses returned by the search.

2. BLOCK PROCESSING

In developing our algorithms, we have decided to work with commercially distributed books on tape (analog recordings of well known novels). Half of each recording is used as training data and the other half for testing and we use an optical character recognizer to read the accompanying texts. Since this data is not segmented into sentences we have designed our training and recogniton algorithms to work with chunks of data of arbitrary size. This means that the data has to be processed in blocks which can fit comfortably into memory.

The approach that we have taken to this problem in the training phase is described in a companion paper [24]. In the recognition phase, we use an A* search in each block which is similar to the isolated word recognition algorithm except insofar as word boundaries are not known in advance and a trigram language model is used in the scoring procedure. As in the isolated word case, an admissible heuristic is obtained by means of an initial Viterbi search through a graph which imposes triphone phonotactic constraints on phone strings. The A* search generates a list of theories (partial phonemic transcriptions together with word histories) for the speech data up to the end of the block[3]. As soon as the list of theories for the current block has been obtained, the block is swapped out of memory and the search of the next block begins using this list to initialize the stack.

This list of theories plays the same role as the beam used in a time synchronous Viterbi search. The Markov property of the trigram language model allows us to merge theories that have identical recent pasts but different remote pasts so the number of theories that have to be generated at the end of each block (the 'beam width') can held fixed without running the risk of losing the optimal theory. In order to pursue the search in subsequent blocks, the only information needed concerns the recent pasts of these theories. By logging the information concerning the remote pasts to disk we are able to ensure that the memory required to recognize a file is independent of its length (instead of increasing exponentially with the length of the file as would be necessary without merging and block processing).

For the last block in a file it is only necessary to generate a single recognition hypothesis and, once the last block has been processed, the transcription of the entire utterance can be obtained by back-tracking. The recognition

[1] This work was supported by the Natural Sciences and Engineering Research Council of Canada

[2] Also with Bell-Northern Research, Montreal, Canada

[3] More precisely, each of the theories generated has the property that all of the hypothesized end points for the third-to-last phoneme in the partial phonemic transcription are beyond the end of the block. The partial phonemic transcription need not end at a word boundary.

algorithm can therefore be viewed globally as a beam search and locally as an A* search.

3. THE HEURISTIC

Broadly speaking, an A* search of the data in a block proceeds as follows. At each iteration of the algorithm, there is a sorted list (or 'stack') of theories each with a heuristic score. This heuristic score is calculated by combining the exact likelihood score of the speech data accounted for by the theory (using acoustic HMMs and the language model) with an overestimate of the score of the remaining data on the optimal extension of the theory permitted by the lexicon and the language model. The theory with the highest heuristic score is expanded, meaning that for each of the one-phoneme extensions permitted by the lexicon the heuristic score of the extended theory is calculated and the extended theory is inserted into the stack at the appropriate position. This process is iterated until sufficiently many theories satisfying a suitable termination criterion have been generated.

Our strategy for overestimating acoustic scores is essentially the same as in the isolated word case, that is, we conduct an exhaustive search backwards in time through a phonetic graph which imposes triphone phonotactic constraints on phoneme strings. This graph is specified as follows and is denoted by G^*.

(i) *Nodes:* there is one node for every possible diphone fg

(ii) *Branches:* for every legitimate triphone fgh (that is, a triphone that can be obtained by concatenating the phonemic transcriptions of words in the dictionary) there is a branch from the node corresponding to the diphone fg to the node corresponding to the diphone gh

(iii) *Branch Labels:* if fgh is a legitimate triphone then the branch from the node fg to the node gh carries the label f.

4. SEARCHING A BLOCK

In order to search a block extending from times T_1 to T_2, we first construct the hidden Markov model corresponding to the graph G^* [16] and, for a suitably chosen positive integer Δ, we perform a Viterbi search backwards in time through this HMM from time $T_2+\Delta$ to time T_1. (The condition used to determine the parameter Δ is given below.) The boundary condition used to initialize the search is that the backward probability at every state in the model at time $T_2+\Delta$ is 1. For each node n in G^* and each $t = T_1-1,\ldots,T_2+\Delta-1$ we thus obtain the Viterbi score of the data in the interval $[t+1, T_2+\Delta]$ on the best path in G^* which leaves n at time t and is subject to no constraints on the state in the model occupied at time $T_2+\Delta$; denote this quantity by $\beta_t^*(n)$.

Suppose we are given a partial phonemic transcription $f_1\ldots f_k$. Let n be the node corresponding to the diphone $f_{k-1}f_k$ and for each time t, let $\alpha_t(f_1\ldots f_{k-2})$ denote the Viterbi score of all of the data up to time t (starting from the beginning of the utterance) for the truncated transcription $f_1\ldots f_{k-2}$. Since $\beta_t^*(n)$ is the Viterbi score of the data in the interval $[t+1, T_2+\Delta]$ on the best path in G^* which leaves n at time t and the construction of G^* constrains this path to pass first through a branch labelled f_{k-1} and then through a branch labelled f_k, it is reasonable to estimate the endpoint of the phoneme f_{k-2} as $\operatorname{argmax}_t \alpha_t(f_1\ldots f_{k-2})\beta_t^*(n)$. In the case of clean speech and speaker-dependent models, this estimate turns out to be exact almost all of the time [16] but it is safer to hypothesize several end points (for instance by taking the five values of t for which $\alpha_t(f_1\ldots f_{k-2})\beta_t^*(n)$ is largest).

A stack entry (theory) θ is a septuple $(\mathbf{w}, \mathbf{f}, m, n, \sigma, \{\alpha_t\}, S)$ where

1. $\mathbf{w} = w_1\ldots w_n$ is a word history.
2. $\mathbf{f} = f_1\ldots f_k$ is a partial phonemic transcription which may extend into a word following w_n (but there are no complete words after w_n in the partial transcription $\mathbf{f}$)
3. m is a node in the lexical tree [16] corresponding to the part $\mathbf{f}$ which extends beyond w_n, if any; m is the root node of the lexical tree otherwise
4. n is the node in the graph G^* which corresponds to the diphone $f_{k-1}f_k$
5. σ is the current state of the trigram language model; there are three possibilities depending on whether the word following w_n is predicted using a trigram distribution $P(\cdot|w_{n-1}w_n)$, a bigram distribution $P(\cdot|w_n)$ or a unigram distribution $P(\cdot)$
6. for each endpoint hypothesis t, α_t is the Viterbi score of the data up to time t against the model for the truncated transcription $f_1\ldots f_{k-2}$
7. S is the heuristic score which is given by

$$S = P(\mathbf{w}) \max_t \alpha_t(f_1\ldots f_{k-2})\beta_t^*(n)$$

where $P(\mathbf{w})$ is the probability of the word string $\mathbf{w}$ calculated using the trigram language model.

The reason why both $\mathbf{w}$ and $\mathbf{f}$ have to be specified is that different words may have the same transcription and different transcriptions may correspond to the same word. Obviously it is redundant to specify m, n and σ in addition to $\mathbf{w}$ and $\mathbf{f}$ but it is convenient to do so.

A stack entry is said to be *complete* if all of its hypothesized endpoints are to the right of T_2. The parameter Δ is determined empirically by the condition that the exact endpoint of a complete stack entry should always be included among the hypothesized endpoints. (Since it is not actually possible because of memory limitations to carry around sufficient information with each theory to be able to generate its segmentation, we test this condition by verifying that the acoustic score of the global transcription found by the recognizer of the data in each file is the same as the score found by the training program when it is run with this transcription.)

At the start of the search, the stack is initialized using the list of theories generated by searching the previous block (ending at time T_1). Each of these has the property that all of its hypothesised endpoints are to the right of T_1, so the speech data prior to the beginning of the current block is no longer needed. The search terminates when sufficiently many complete theories have been generated at which point the next block is swapped into memory and a new search begins.

The Markov property of the trigram language model enables us to merge theories that have identical recent pasts but different remote pasts. Specifically, suppose we have two theories $\theta = (\mathbf{w}, \mathbf{f}, m, n, \sigma, \{\alpha_t\}, S)$ and $\theta' = (\mathbf{w}', \mathbf{f}', m', n', \sigma', \{\alpha_t'\}, S')$ such that $m = m'$, $n = n'$ and $\sigma = \sigma'$. (In this case we will say that θ and θ' are equivalent.) The future extensions of both theories which best account for the data starting at any given time (subject to lexical and language model constraints) will be identical. Thus if it happens that t is on the list of hypothesized endpoints for both theories and

$$P(\mathbf{w}')\alpha_t' < P(\mathbf{w})\alpha_t$$

then we can remove t from the hypothesis list for the second theory without running the risk of losing the optimal path. In practice, the condition $n = n'$ means that the list of hypothesized endpoints for both theories will be the same (except in very rare cases). Furthermore, if this inequality holds for one such t then it is typically because the first theory gives a better fit to the remote past than the second theory; hence it will usually be the case that if the inequality holds for one t then it will hold for all t and the second theory can be pruned away completely.

We can take advantage of this fact to speed up the A* search by maintaining a list of 'merge buckets' consisting of all the equivalence classes of theories encountered in the course of the search. Associated with each equivalence class we have an array of forward scores $\{A_t\}$ which is updated throughout the search. For each t, A_t is defined to be $\max_\theta P(\mathbf{w})\alpha_t$ where θ extends over all theories $(\mathbf{w}, \mathbf{f}, m, n, \sigma, \{\alpha_t\}, S)$ in the given equivalence class that have been encountered so far (in the course of searching the current block). When a new theory $\theta' = (\mathbf{w}', \mathbf{f}', m', n', \sigma', \{\alpha_t'\}, S')$ in this equivalence class comes to be inserted into the stack we can test to see if the inequality

$$P(\mathbf{w}')\alpha_t' < A_t$$

holds for each hypothesized endpoint t. If it does, then we can prune this endpoint hypothesis before entering the theory into the stack; if not, then A_t is updated and the endpoint hypothesis has to be retained.

We have not been able to implement this scheme fully because of memory limitations. In practice, we only invoke merging when a word boundary is hypothesized so the only merge buckets generated in the course of the search are those which correspond to theories for which m is the root node of the lexical tree. (However, before starting the search we prune the list of hypotheses generated by searching the previous block by merging at arbitrary phoneme boundaries and we use this pruned list to initialize the stack.)

5. EXPERIMENTAL RESULTS

For development purposes we have been using recordings on commercially distributed analog cassettes of a male speaker reading *White Fang* (by Jack London) and a female speaker reading *Washington Square* (by Henry James). The data was digitized at 16 kHz and the acoustic features used for our experiments a set of eight static and seven dynamic mel-based cepstral

coefficents (calculated every 10 ms). Phonetic HMMs for each of the speakers were trained using the algorithm described in [24] (with training set sizes of 14,000 words in the case of *White Fang* and 16,000 words in the case of *Washington Square*).

In the recognition experiments we used a dictionary of 60,000 words, containing an average of 2.2 transcriptions per word, which was augmented to include all of the words in both books. (1.5% of the words were missing in each case. The majority of the missing words were inflected forms of words already in the dictionary.) The language model used was a trigram language model trained on 60,000,000 words of newspaper text. The test set perplexities calculated with this model were 1,743 in the case of *White Fang* and 749 in the case of *Washington Square*[4].

Our principal results are as follows:

Book (Sex)	Test Set	Accuracy (CI)	Accuracy (CD)
WF (M)	730	58%	68%
WS (F)	585	75%	80%

Table 1 Recognition results on two books.

The second column in this table gives the test set sizes in words; the third and fourth columns give the recognition accuracies for experiments performed with context-independent (CI) and context-dependent (CD) HMMs; the accuracies are calculated using the formula

$$\frac{N - (\text{Substitutions} + \frac{1}{2}[\text{Deletions} + \text{Insertions}])}{N}$$

where N is the size of the test set. The context-independent models consisted of a collection of 41 mixture HMMs (one per phoneme) each having a single covariance matrix common to all mixture components in the model. The context-dependent models consisted of 137 mixture HMMs with covariance matrices tied among the allophones of each phoneme. Allophone clustering was performed by means of a decision tree [23] (but no attempt was made to model cross-word effects).

The recognizer was configured as follows. In order to keep the size of the stack within reasonable bounds, we had to use a block advance of only 10 frames (1 frame = 10 ms). The parameter Δ was set at 140 frames (but it turns out that we could have taken Δ to be 50 frames without occurring any penalty in accuracy). The maximum number of entries in the stack was set 60,000. (Whenever this figure was attained, the size of the stack was cut back to 30,000). The number of theories passed from one block to the next was 3,000. Using context-independent models, the CPU time required to run the experiments on a HP 720 workstation was 120 times real time.

The results in Table 1 are the fruit of a long series of experiments. In the case of *White Fang*, the recognition rate on our very first experiment using context-independent models was 40%; screening the training data and smoothing the estimate of the covariance matrix of the silence model gave 52%; treating compound proper names such as *White Fang*, *Lip Lip* and *Gray Beaver* as single words rather than sequences of two words for the purpose of calculating language model scores increased the accuracy to 54%; some alignment errors made by the training algorithm were corrected by detecting long silences in a pre-processor, giving an accuracy of 58%; finally, context-dependent models increased the accuracy to 68%.

Two experiments that did *not* lead to improvements in performance are worth mentioning. We found that dramatic reductions in the perplexities of the test sets could be obtained by using frequency counts of words in the training sets to smooth the language model statistics. The perplexity was reduced from 1,743 to 749 in the case of *White Fang* and from 576 to 347 in the case of *Washington Square*. However there was no significant reduction in the recognition error rate in either case. Secondly, the use of constraints on phoneme durations which proved so effective in the case of isolated word recognition [25] has not contributed to improving the accuracy (although it does speed up the search).

[4] The test set perplexities P were calculated in the usual way, using the formula

$$P = \Pr(w_1 \dots w_n)^{\frac{1}{n}}$$

where $w_1 \dots w_n$ is the string of words in the test set. This way of calculating perplexities suffers from the the disadvantage that the scoring mechanism for words not included in the language model's vocabulary is somewhat arbitrary. However, this fact does not seem to account for the very high perplexity of WF compared to WS since the percentage of out-of-vocabulary words was 1.4% in the case of WF and 2.2% in the case of WS.

6. FUTURE WORK

We have reported the results of some pilot experiments on continuous speech recognition using a 60,000-word vocabulary and a trigram language model. Obviously, these results are only significant in so far as they indicate the types of improvement that will be needed if speech recognition on this scale is to become a practical reality.

We expect that the biggest payoff in recognition accuracy will come from greatly increasing the allophone model inventory. Given the size of the vocabulary, the robustness problem is obviously a major issue. (The number of triphones in the dictionary is a about 17,000; this number increases by a factor of about two when triphones spanning word boundaries are included.) We are experimenting with an approach to generalized triphone modelling using multiple templates to score every possible triphone. These templates are context dependent models where context is described specifying the values of phonetic features on the right and the left in varying degrees of precision. Included among the templates are context-independent models and triphone models (in the case where the triphone is observed in the training data) as well as a large number of intermediate models.

In [21] we described how an admissible heuristic for searching with allophone models could be calculated without changing the topology of the graph G^*. However we have decided not to pursue this since the evidence in favour of a two-pass approach to large-scale speech recognition problems now seems to be quite compelling [22]. Accordingly, we are redesigning the search strategy so that in the first pass the data in a block is searched using a coarse set of allophone models to match surface form phonemic transcriptions and a coarse language model. This search (which is essentially the same as the algorithm reported here) generates a list of partial recognition hypotheses for the data in the block, each hypothesis consisting of a short word string (possibly empty) together with a partial transcription of a word which extends beyond the end of the block. These hypotheses are rescored in the second pass using fine allophone models, a fine language model and a phonological component which checks the consistency of the surface form transcriptions hypothesized by the first pass. (In the new algorithm, the first pass treats surface form transcriptions of the words in each hypothesis as ocurring in free variation; the second pass rejects any hypothesis whose surface form transcription cannot be obtained by the application of phonological rules to the base form transcriptions of the words in the hypothesis.)

We have to resolve the question of whether a fast match ought to be incorporated into the search. The idea of a fast match is to conduct a crude search of the entire lexical tree every time a word boundary is hypothesized; the algorithm we have presented can be viewed as maintaining several copies of the lexical tree in memory and abandoning the search of a given copy of the lexical tree whenever the heuristic indicates that it would be more promising to search another. This approach was motivated by the need to avoid the computational overhead of running a fast match to completion every time it is invoked, but the memory requirements are so great that we were constrained to work with a very small block advance. This problem will be alleviated by using a coarse language model in the first pass, but we are also experimenting with a an approximate fast match which uses information extracted from the search of the graph G^* to score partial transcriptions at a cost of a very small number of multiplications per phoneme.

Since we are incorporating a phonological component to convert base forms to surface forms and then scoring surface forms using context-dependent allophone models, the search problem becomes quite complex even in the training phase (especially since we want to be able to handle unsegmented training files of arbitrary length). To deal with this, we are designing the recognition search strategy so that it can accomodate an abstract language model; this enables us to perform the Viterbi alignment of the data in a training file by supplying a language model constructed from the words in the training script.

7. REFERENCES

[1] Averbuch A., "Experiments with the Tangora 20,000 word speech recognizer," *Proc. ICASSP 87*, pp. 701-704, 1987.

[2] Bahl, L.R. et al. "Large Vocabulary Natural Language Continuous Speech Recognition", *Proc. ICASSP 89*, pp. 465–467, 1989.

[3] Bahl, L.R., De Gennaro S.V., Gopalakrishnan, P.S., Mercer, R.L., "A fast approximate acoustic match for large vocabulary speech recognition", personal communication.

[4] Bahl, L.R., Bakis, R., de Souza, P.V., Mercer, R.L., "Obtaining candidate words by polling in a large vocabulary speech recognition system", *Proc. ICASSP 88*, pp. 489–492, 1988.

[5] Fissore, L., Laface, P., Micca, G., Pieraccini, R., "Lexical access to large vocabularies for speech recognition", *Proc. ICASSP 89*, pp. 1197–1213, 1989.

[6] Ney, H., Haeb-Umbach, R., Tran B.-H., Oerder, M., "Improvements in beam search for 10000-word continuous speech recognition", *Proc. ICASSP 92*, pp. I-9–I-12, 1992.

[7] Sagayama, S., "A matrix representation of HMM-based speech recognition algorithms", *Proc. Eurospeech 91*, pp. 1225–1228, 1991.

[8] Soong, F.K., Huang, E.-F., "A tree-trellis based fast search for finding the N best sentence hypotheses in continuous speech recognition", *Proc. ICASSP 91*, pp. 705–708, 1991.

[9] Zue, V. et al., "Integration of speech recognition and natural language processing in the MIT VOYAGER system", *Proc. ICASSP 91*, pp. 713–716, 1991.

[10] Austin, S., Schwartz, R., Placeway, P., "The forward-backward search algorithm", *Proc. ICASSP 91*, pp. 697–700, 1991.

[11] Jelinek, F., "A fast sequential decoding algorithm using a stack", *IBM Journal of Research and Development*, **13**, pp. 675–685, 1969.

[12] Jelinek, F., "Continuous Speech Recognition by Statistical Methods", *Proc. IEEE*, **64**, 1976.

[13] Seitz, F., Gupta, V., Lennig, M., Kenny, P., Deng, L., and Mermelstein, P. "Dictionary for a very large vocabulary word recognition system," *Computer, Speech and Language*, **4**, pp. 193–202, 1990.

[14] Nilsson, N., "Principles of artificial intelligence," Tioga Publishing Company, 1982.

[15] Gupta, V., Lennig, M., and Mermelstein, P., "Fast search strategy in a large vocabulary word recognizer," J. Acoust. Soc. Am. **84**(6), 2007-2017, 1988

[16] Kenny, P., Hollan, R., Gupta, V., Lennig, M., Mermelstein, P., and O'Shaughnessy, D., "A* - admissible heuristics for rapid lexical access", to appear in *IEEE Transactions on Signal Processing*, November 1992.

[17] Deng, L., Kenny, P., Lennig, M., Gupta, V., Seitz, F., and Mermelstein, P., "Phonemic hidden Markov models with continuous mixture output densities for large vocabulary word recognition," *IEEE Transactions on Signal Processing*, **39**, pp. 1677–1681, 1991.

[18] Kenny, P., Parthasarathy, S., Gupta, V., Lennig, M., Mermelstein, P., and O'Shaughnessy, D., "Energy, Duration and Markov models", *Proc. Eurospeech 91*, pp. 655–658, 1991.

[19] Lennig, M., Gupta, V., Kenny, P., Mermelstein, P., O'Shaughnessy, D., "An 86,000-Word Recognizer Based on Phonemic Models", *Proc. DARPA Speech and Natural Language Workshop*, pp. 391–396, 1990.

[20] Paul, D., "Algorithms for an optimal A* search and linearizing the search in the stack decoder", *Proc. ICASSP 91*, pp. 693–696, 1991.

[21] Kenny, P., Hollan, R., Boulianne, G., Garudadri, H., Lennig, M. and O'Shaughnessy, D., "An A* algorithm for very large vocabulary continuous speech recognition", *Proc. DARPA Speech and Natural Language Workshop*, February 1992.

[22] Schwartz, R., Austin, A., Kubala, F., Makhoul, J., Nguyen, L., Placeway, P., Zavaglios, G., "New Uses for the N Best Sentence Hypotheses within the Byblos Speech Recognition System", *Proc. ICASSP 92*, pp. I-1–I-4, 1992.

[23] Bahl, L., de Souza, P., Gopalakrishnan, P., Nahamoo, D., Picheny, M., "Decision Trees for Phonological Rules in Continous Speech", *Proc. ICASSP 91*.

[24] Boulianne, G., Kenny, P., Lennig, M., O'Shaughnessy, D., Mermelstein, P., "HMM training on unconstrained speech for large vocabulary continuous speech recognition", these proceedings.

[25] V.N. Gupta, M. Lennig, P. Mermelstein, P. Kenny, F. Seitz and D. O'Shaugnessy, "Using phoneme duration and energy contour information to improve Large Vocabulary Isolated Word Recognition", *Proc. ICASSP 91*.