A One Pass Decoder Design For Large Vocabulary Recognition

J.J. Odell, V. Valtchev, P.C. Woodland, S.J. Young

Cambridge University Engineering Department
Trumpington Street, Cambridge, CB2 1PZ, England

ABSTRACT

To achieve reasonable accuracy in large vocabulary speech recognition systems, it is important to use detailed acoustic models together with good long span language models. For example, in the Wall Street Journal (WSJ) task both cross-word triphones and a trigram language model are necessary to achieve state-of-the-art performance. However, when using these models, the size of a pre-compiled recognition network can make a standard Viterbi search infeasible and hence, either multiple-pass or asynchronous stack decoding schemes are typically used. In this paper, we show that time-synchronous one-pass decoding using cross-word triphones and a trigram language model can be implemented using a dynamically built tree-structured network. This approach avoids the compromises inherent in using fast-matches or preliminary passes and is relatively efficient in implementation. It was included in the HTK large vocabulary speech recognition system used for the 1993 ARPA WSJ evaluation and experimental results are presented for that task.

1. INTRODUCTION

Hidden Markov Models (HMMs) have been used successfully in a wide variety of recognition tasks ranging from small isolated word systems assisted by heavily constrained grammars to very large vocabulary unconstrained continuous speech systems. Part of the success of HMMs is due to the existence of computationally efficient algorithms for both the training of the models (the Baum-Welch algorithm) and for the decoding of unknown utterances (the Viterbi algorithm). However, as recognition tasks have become more complex, the decoding process has become more difficult due to the increasing size of network needed in a conventional Viterbi decoder. In particular, using cross-word triphones or long span language models can lead to order of magnitude increases in the size of a static network.

A variety of schemes have been proposed to reduce the computation required for recognition [2,7]. Most make use of fast-matches or preliminary passes using simplified acoustic or linguistic models to constrain the search space for a final pass that uses the most detailed and accurate models available. Unfortunately, these preliminary matches can introduce errors that subsequent passes are unable to correct. If the first pass could use the best acoustic and language models available it would allow greater constraints to be placed on the search space without increasing the error rate. This paper describes a scheme which allows this through the use of a dynamically built tree-structured network. This approach avoids the compromises inherent in using fast-matches or preliminary passes and is both simple and efficient to implement.

The remainder of this paper is organised as follows. Section 2 discusses the main features of conventional time-synchronous Viterbi decoding and some of the ways in which it can be improved. In section 3, a one-pass decoder that implements a beam-pruned Viterbi search through a tree-structured dynamic network is then described. Section 4 presents some experimental results on the Wall Street Journal task and, finally, section 5 presents our conclusions on this work.

2. VITERBI DECODING

2.1. The Standard Viterbi Search

The standard method of implementing a Viterbi search for decoding speech into words is to build a re-entrant network of HMM *phone instances*. An instance of each word in the vocabulary is then made from a concatenated sequence of phone instances and the end of each *word instance* is linked, according to a language model or grammar, back to the start of the word instances. Decoding uses a time-synchronous Viterbi search through this network in which partial state/frame alignment paths are maintained and extended in parallel using the Viterbi criterion (i.e. the principle of dynamic programming). In our work, each path is denoted by a *token* so that path extension can be represented simply by propagating tokens through the network [12].

A complete Viterbi search is admissible and will not make any search errors whereby the decoder fails to correctly find the most likely sequence of models for a given utterance. Unfortunately, because of the size of these networks, a complete search is not computationally feasible even for moderately sized tasks. Consequently it is

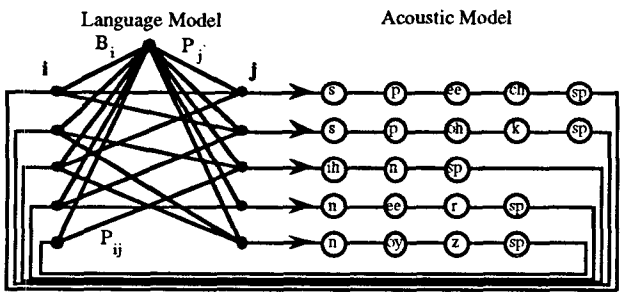

Figure 1: A Static Network for Viterbi Decoding with a Back-off Bigram Language Model

necessary to reduce (or prune) the search space to speed-up the search. A common strategy for doing this is to use a *beam search* in which only paths whose likelihood falls within a fixed beam width of the mostly likely path are considered for extension. As an example of this kind of approach, Fig. 1 shows a typical static network structure for word recognition using monophone models and a bigram-backoff language model [4].

2.2. The Limitations of Static Networks

Recent uses of HMMs in very large vocabulary tasks have begun to show the deficiencies of the static network architecture. Such systems typically have a large number of context-dependent models and they use unconstrained long span statistical language models. As is clear from Fig. 1, the number of individual phone instances scales linearly with the size of the vocabulary whilst the number of possible cross links, for the bigram case, scales with the square of the number of words. For the trigram case, matters are even worse and each word pair that has trigram language model probabilities for subsequent words must be explicitly represented in the network. Depending on the size of the language model, this could result in a dramatic increase in the size of the network.

Similarly, increasing the number of phone models used can lead to correspondingly large increases in the size of network needed to decode them. For example, if cross-word context dependent triphone models are used during recognition, the size of the recognition network increases substantially. Rather than having a single model transcript for each word, the initial (and final) phone must be replaced by one of a set of models. The size of the set depends on which phones may occur at the end of the preceding (or start of the following) word. Normally this is around the same size as the phone set and so an average word will require almost 100 model instances.

To give a specific example, the Dragon Wall Street Journal Pronunciation Lexicon Version 2.0 dictionary contains 19979 words with 21875 pronunciations. Using a set of 44 distinct phone models, this leads to a total of approximately 150,000 phone instances. If cross-word triphones are used, the number of required phone instances rises to around 1,800,000. Finally, the standard WSJ 20k trigram language model has approximately 740,000 word-pairs with trigram probabilities. If this is used in a static network structure, the number of needed phone instances rises to around 24,000,000.

2.3. Stack Decoders

Many of the pressures leading to increased network size result directly from the breadth-first nature of the time-synchronous Viterbi search. An obvious alternative is therefore to use a depth-first scheme. For example, stack decoders have been used with some success [5,9]. In a stack decoder, hypotheses are advanced one word at a time and hence long span language models and context dependency can be supported without the need for large networks. However, the indeterminacy of word boundaries in continuous speech and the variety of possible right contexts at the ends of words means that, in practice, a stack decoder must use a preliminary fast-match to reduce the number of path extensions considered. This is a major drawback since fast-matches can be a serious source of search errors.

2.4. Reducing Computation

If a time-synchronous beam search is to be used, then the pruning strategy needs to be as effective as possible. In practical systems, variable width and multiple beam schemes are more effective than using a single fixed beam width[1,6]. For example, a higher degree of uncertainty exists at the start of words than at the end. Hence, using an additional word-end beam can result in substantially fewer active models without significantly increasing the error rate.

The use of such an additional beam can also be justified on the basis of the language models employed. Although the probabilities in a language model may vary by several orders of magnitude, the probability of a particular word actually varies much less. For example, in the standard 5k closed vocabulary bigram WSJ language model, the highest and the lowest probabilities vary by a factor of 10^7. However, on average over 99.5% of the probabilities for a particular word lie within a factor of 100 of each other. This is due to the very heavy reliance on the back-off component of the language model. It means that few search errors will be introduced by only propagating word-end tokens for which the likelihood is within

a factor of 100 or so of the most likely word-end token. This implies that the word-end beam width can be much narrower than the width of the normal beam.

2.5. Tree Structuring

Since the uncertainty in decoding speech is much higher at the start of words than at the ends, it follows that the majority of the computation is expended on the first few phones of each word [8]. For very large vocabularies, a tree structured network in which the words that share common initial phone sequences share model instances, achieves the dual aim of reducing the size of the network as well as the computation required to decode it. Hence, a tree-structured organisation is highly desirable.

When using a tree-structured organisation, it is important to ensure that, although word identity is not explicitly known until the end of the word is reached, the application of the language model is not delayed until then. If this is not done, the relaxation of the constraints on the search can offset the computational savings and will require the use of larger beam widths to avoid search errors [3].

3. A ONE-PASS DECODER

From the discussion in the previous section, it is clear that the key features of a successful single-pass decoding scheme are the ability to incorporate cross-word triphones and long span language models whilst keeping both the computational time and space requirements within acceptable bounds. To do this, it is clearly necessary to tree-structure the recognition network and to apply tight and efficient pruning. To make this possible, the concept of a static re-entrant network must be abandoned. Instead, a non re-entrant tree-structured network must be used with a new copy of the tree being replicated at every word end. To make this fit in available memory, the network must be grown dynamically *on-the-fly* and once phone instances fall outside of the beam, the corresponding nodes must be reclaimed.

This section describes such a decoder. It uses the token passing paradigm to implement a beam pruned Viterbi search through a dynamic tree structured network of HMM instances.

3.1. Network Structure

Due to the tree-structured nature of the network and the possibility that two words may have exactly the same phonetic realisation (and will therefore share all their models) it is necessary to have some point at which the identity of a word becomes unique. Consequently the recognition network consists of two types of nodes.

- HMM instances. These represent an actual phone from the dictionary and are linked to a physical HMM (the identity of which may depend on the context). The HMM is used to calculate the acoustic likelihood of that phone instance and the network node holds the tokens that store the associated likelihoods and paths.
- Word-ends. These are linked to a particular word and are the points where the language model likelihoods are added to the acoustic likelihoods.

The HMM instances are connected in a simple tree structured network in which each model has a specific predecessor but may have many followers. Word-end nodes are also linked to each other when token recombination or domination can occur(see below). Fig 2 shows a fragment of a typical network configuration.

Each node in the network has an associated language model probability. This is added to the token likelihood to give the combined likelihood that is used for pruning purposes. At word-ends, the language model can provide the exact probablitiy for the word given its history. However, HMM phone instances can be shared by many words and hence only an approximation for the language model probability can be used within a word. Therefore, until the word identity is uniquely defined, the highest language model probability of all words that share the instance is used. This guarantees that the probability used is always an exact upper bound on the actual probability and this helps to minimise search since it can never increase through a word. Using the exact upper bound allows the tightest beam widths to be used without introducing search errors. The overhead for dynamically constructing the network and for using the exact language model for calculating the upper bound on likelihoods is relatively small and rarely exceeds 20% of the total computational load.

3.2. Recombination and Dominance

When a static network is used for recognition, tokens recombine at the start of each word and only one survives and is propagated into the word. These recombination points are where the Viterbi criterion is applied to decide which of the set of tokens is part of the maximum likelihood path. In general, three conditions must be fulfilled to allow tokens to recombine

- The following network must be identical in structure.
- Corresponding network nodes must have the same acoustic likelihoods.

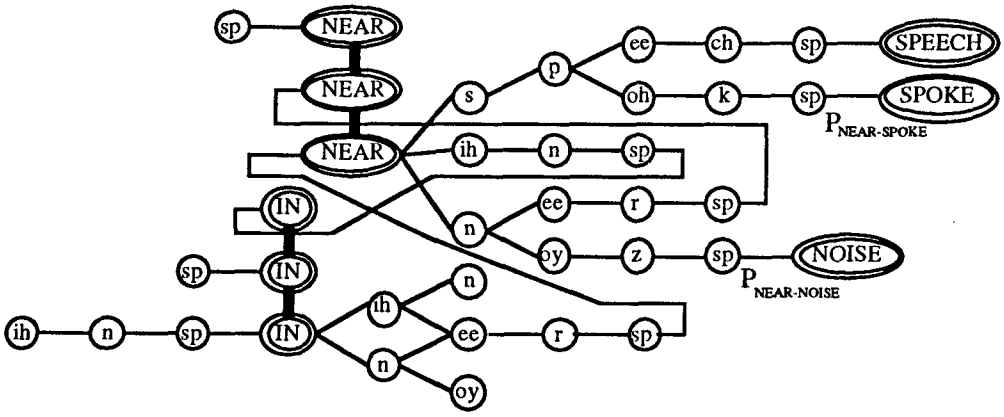

Figure 2: A Fragment of a Tree Structured Network

- Corresponding network nodes must have the same language model likelihoods.

As word-ends are created, they are linked to any existing word-ends that meet these conditions to form *dominance* chains. The Viterbi criterion means that only the most likely token in the word-end nodes on one of these chains will form part of the most likely path. Due to the fact that the network is dynamically constructed, there are two ways in which this can be applied.

- Recombination. The word-end nodes on a dominance chain share a single set of successor nodes. The most likely token on the dominance chain is then propagated into the following models as in a standard Viterbi search. Note that this implies that the token itself must contain traceback information since it may have come from one of a number of paths.
- Dominance. Each word-end on the dominance chain can have its own set of successor nodes. However only the most likely word-end on the chain is allowed to create any successor nodes. Thus, each network node will have a unique history and so traceback information can be held at the network level rather than at token level.

At first it may seem that token recombination is the most sensible course since domination can lead to the creation of multiple copies of network structure when a single one would suffice. In practice, however, this happens very rarely and holding the traceback information at the network level rather than in each token means that each HMM instance can be more compact. There is thus a trade-off between memory usage and computation required. In practice, using domination rather than recombination leads to around 5-10% more active models (and hence computation) but results in a 10-20% reduction in the memory required to hold the network.

The token recombination method does have one distinct advantage. As explained later in this paper, it is possible to produce a lattice of word hypotheses rather than the single best sentence for very little extra computational effort if each token contains traceback information.

3.3. Network Construction

The network is constructed dynamically and nodes are only created when they will fall into the beam and are destroyed as soon as they leave the beam. In this case, pruning becomes doubly important since it controls not only the amount of computation used but also the amount of memory required.

Network growth occurs during model-external token propagation. The network is extended if the combined likelihood of a token has no following node and falls within the beam. Since the combined likelihood of the token after propagation into the newly created node will depend on the language model likelihood of that node, this combined likelihood is calculated in advance to prevent network nodes being created unnecessarily. The additional computation involved in doing this is much less than the memory creation/disposal overhead would otherwise be.

When nodes are constructed, the identity of the phone and the word in which it occurs, together with its context (at both the phone and the word level), the position of surrounding word boundaries and the gender of the speaker may all be used to select which HMM will be used for that node. This allows the use of function word specific, position dependent, gender dependent and long

distance context dependent phonetic models.

Pruning is at the model level and occurs in several ways.

- Blocking. Tokens that fall outside of the beam are blocked to prevent network growth from occurring. Word-end nodes have their own separate beam which is also checked before network growth from a word-end node occurs.
- Erasure. A node that falls outside the beam and which has no predecessors is erased. The space allocated to the node is freed, the node is removed from the network and will never be re-created.
- Deletion. A node that falls outside of the beam and which has no followers is deleted. This involves freeing the node in such a way that it may be re-created if it comes back into the beam.
- Halting. A node that falls outside the beam which has predecessors and followers is halted. The node is not removed from the network (since it would be difficult to re-create and link back into the correct place) but is marked as inactive. Internal token propagation does not occur for inactive nodes and so, although it is using memory, it requires little computation.

All other nodes fall within the beam and are active. Both internal and external token propagation will occur for these models. The computational load varies approximately linearly with the number of active models.

3.4. N-Best Lattices

During decoding there are multiple copies of each word active and it is therefore possible to generate multiple sentence hypotheses with little additional computation. This is implemented by linking the tokens that occur in chained word-end nodes and propagating the most likely token (which is the head of this list) into following nodes exactly as before. The multiple hypotheses can be recovered by descending the list at each boundary at the end of the utterance. This will not generate exact solutions for any but the best path since it implicitly assumes that the start time of each word is independent of all words before its immediate predecessor. However, it has been shown that this is a reasonable assumption and has little effect on overall lattice accuracy [10].

3.5. One-Pass Algorithm

The above one-pass decoder strategy has been implemented in a simple three step algorithm:-

- Create a single *sentence_start* node for each gender-dependent model set.
- For each frame of input
  - Prune from the network all models for which the combined acoustic and language model likelihood falls outside of the beam.
  - Perform token propagation within each HMM instance and find the top of the beam for the next time step.
  - Perform token propagation between nodes extending the network when this is necessary.
- Find the most likely *sentence_end* node and trace-back to find the resulting word sequence.

4. EXPERIMENTAL RESULTS

Experiments have been performed on both 5k and 20k Wall Street Journal tasks. The WSJ systems used training data from the SI-84 and SI-284 test sets, and the pronunciations from the Dragon Wall Street Journal Pronunciation Lexicon Version 2.0 together with the standard bigram and trigram language models supplied by MIT Lincoln Labs. Some locally generated additions and corrections to the dictionary were used and the stress markings were ignored resulting in 44 phones plus silence. Data preparation used the HTK Hidden Markov Model Toolkit [13]. All speech models had three emitting states and a left-to-right topology and used continuous density mixture Gaussian output probability distributions tied at the state level using phonetic decision trees [14]. The decoder enforced silence at the start and end of sentences and allowed optional silence between words. These systems achieved the lowest error rates reported for the November 1993 WSJ evaluations on the H1-C2, H2-C1 and H2-P0 and the second lowest error rate on H1-P0. Further details about these systems can be found in [11].

Table 1 gives details of decoder performance for the various tasks. All figures quoted are for the beam widths used in the evaluation tests. The required computation scales with the number of active models per frame (and the number of frames in the test set) and on an HP735 decoding the 5k gender dependent cross-word systems required approximately 10 minutes per sentence whilst the 20k systems took about 15 minutes per sentence (on average). As the table shows, the computation required does not depend on the potential network size since the load for the trigram case is generally less than the corresponding bigram case. This shows that the early application of knowledge can be used to constrain the search in order to offset the computational costs of using the

System Type	Training Data	Task	Number of States, Models & Triphones per gender	Potential Network Size per gender	Average Number of Active Models per frame	Word Error Rate
Word Internal GI	SI84	5k Bigram	3701 / 8087 / 14344	40,000	9600	12.5
Cross Word GD	SI84	5k Bigram	3820 / 15303 / 35633	400,000	21900	8.7
Cross Word GD	SI284	5k Bigram	7558 / 22978 / 35633	400,000	23400	6.8
Cross Word GD	SI284	5k Trigram	7558 / 22978 / 35633	5,000,000	19800	4.9
Cross Word GD	SI284	20k Bigram	7558 / 22978 / 54457	1,800,000	30700	14.4
Cross Word GD	SI284	20k Trigram	7558 / 22978 / 54457	24,000,000	29300	12.7

Table 1: System characteristics for various WSJ tasks.

knowledge. In the bigram case, no reliance is made on the back-off nature of the language model and the computational load will not therefore change when the size of the language model is increased.

5. CONCLUSIONS

One-pass decoding has many advantages over multi-pass decoding if it can be accomplished with a similar amount of computation. This paper has described a method of decoding continuous speech using context-dependent hidden Markov models and long span language models in a single pass. The decoder is relatively simple and efficient and should scale well with increasing size of both the vocabulary and the language models.

6. ACKNOWLEDGEMENTS

The WSJ pronunciation dictionary was provided by Dragon Systems Inc. J. Odell is funded by a SERC studentship and part of this work was funded by SERC grant GR/J10204.

References

1. Alleva F, Hon H, Huang X, Hwang M, Rosenfeld R, Weide R (1993). *Applying SPHINX-II to the DARPA Wall Street Journal CSR Task.* Proc. DARPA Speech and Natural Language Workshop 1992, pp 393-398.
2. Alleva F, Huang X, Hwang M-Y (1993). *An Improved Search Algorithm Using Incremental Knowledge for Continuous Speech Recognition.* Proc. ICASSP'93, Vol II, pp.307-310. Minneapolis.
3. Aubert X, Dugast C, Ney H, Steinbiss V (1994). *Large Vocabulary Continuous Speech Recogniiton of Wall Street Journal Data.* Proc. ICASSP'94 *forthcoming.* Adelaide.
4. Austin S, Peterson P, Placeway P, Schwartz R, Vandergrift J (1990). *Towards a Real-Time Spoken Language System Using Commercial Hardware.* Proc. DARPA Speech and Natural Language Workshop 1990, pp 72-77.
5. Jelinek F, Bahl LR, Mercer RL. *Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech.* IEEE Trans Information Theory, Vol 21, No 3, pp250-256, 1975
6. Lacouture R, Normandin Y (1993). *Efficient Lexical Access Strategies.* Proc. Eurospeech'93, Vol III, pp. 1537-1540. Berlin.
7. Murveit H, Butzberger J, Digalakis V, Weintraub M (1993). *Large-Vocabulary Dictation Using SRI's Decipher Speech Recognition System: Progressive Search Techniques.* Proc. ICASSP'93, Vol II, pp.319-322. Minneapolis.
8. Ney H, Haeb-Umbach R, Tran B-H & Oerder M. (1992). *Improvements in Beam Search for 10000-Word Continuous Speech Recognition.* Proc. ICASSP'92, Vol I, pp. 9-12. San Francisco.
9. Paul D, Necioglu B (1993). *The Lincoln Large-Vocabulary Stack-Decoder HMM CSR.* Proc. ICASSP'93, Vol II, pp.660-663. Minneapolis.
10. Schwartz R, Austin S (1990). *Efficient, High-Performance Algorithms for N-Best Search.* Proc. DARPA Speech and Natural Language Workshop 1990, pp 6-11.
11. Woodland PC, Odell JJ, Valtchev V, Young SJ (1994). *Large Vocabulary Continuous Speech Recognition Using HTK.* Proc. ICASSP'94 *forthcoming*, Adelaide.
12. Young SJ, Russell NH, Thornton JHS (1989) *Token Passing: A Simple Conceptual Model for Connected Speech Recognition Systems.* Technical Report CUED/F-INFENG/TR38, Cambridge University Engineering Dept.
13. Young SJ (1993). *The HTK Hidden Markov Model Toolkit: Design and Philosophy.* TR 152, Cambridge University Engineering Dept, Speech Group.
14. Young SJ, Odell JJ, Woodland PC (1994). *Tree-based State Tying for High Accuracy Acoustic Modelling.* Proc. ARPA Human Language Technology Workshop 1994.