TREE ORGANIZED LEXICONS AND BEAM SEARCH:

IMPLEMENTATIONAL ISSUES

Carlos Alberto Ynoguti
Departamento de Telecomunicações
Instituto Nacional de Telecomunicações (INATEL)
ynoguti@inatel.br

Fábio Violaro
Departamento de Comunicações
Faculdade de Engenharia Elétrica e de Computação
Universidade Estadual de Campinas (UNICAMP)
fabio@decom.fee.unicamp.br

ABSTRACT

The aim of this work is to enlighten some implementational issues on two widely used techniques for reduction in search space for large vocabulary speech recognition systems: tree organized lexicons and Viterbi Beam Search. We show how we can exploit some symmetries of the search space to use these techniques together and achieve a great reduction in computational load and, consequently, in recognition times. Tests showed that recognition times fall down by a factor of 5 using this procedure.

1 INTRODUCTION

In HMM based large vocabulary continuous speech recognition systems, the search for the best word sequence – in a maximum a posteriori sense – is one of the most time consuming tasks. When aligning the locution to be recognized with the reference patterns, we must consider a large number of paths due to vocabulary size and the various alignment possibilities produced by continuous speech [9]. So, to recognize a given spoken sentence, we have to perform an astronomical amount of calculations, and it takes a lot of time even with very fast processors.

Several techniques have been developed to avoid such computational effort and in this work we study two of them: the organization of the pronunciation lexicon in a prefix tree structure [2][3], and a complexity reduction procedure called *Beam Search* [4] that eliminates improbable candidates for the best path in the search space.

We exploit some symmetries of the search space to achieve an extra reduction of the calculations to be made. In fact, we observed a reduction by a factor of 5 in recognition times using the tree organized lexicon system when compared to the linear organized lexicon system.

2 TREE ORGANIZED LEXICONS [8].

For a small vocabulary system, it's sufficient to have a separate representation for each word. This strategy is called a linear lexicon. An example of linear lexicon is shown in Figure 1. For large vocabularies, however, it's useful to organize the lexicon in a prefix tree structure, as depicted in Figure 2.

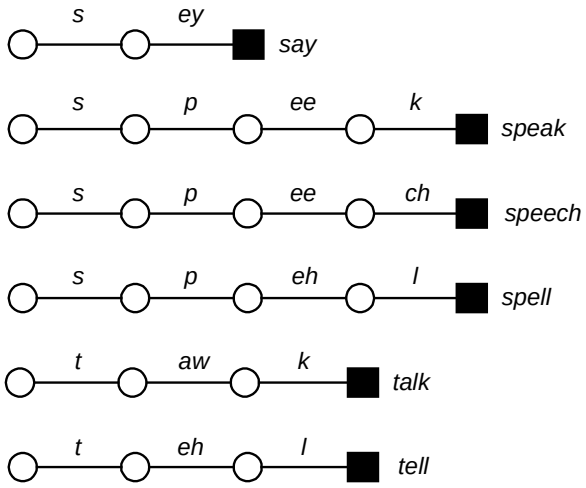


Figure 1. Part of a linear organized lexicon.

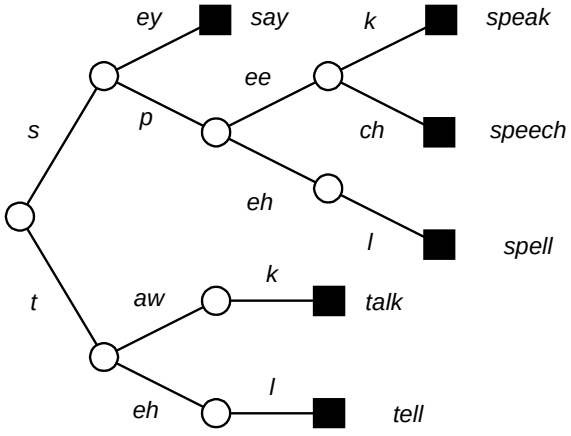


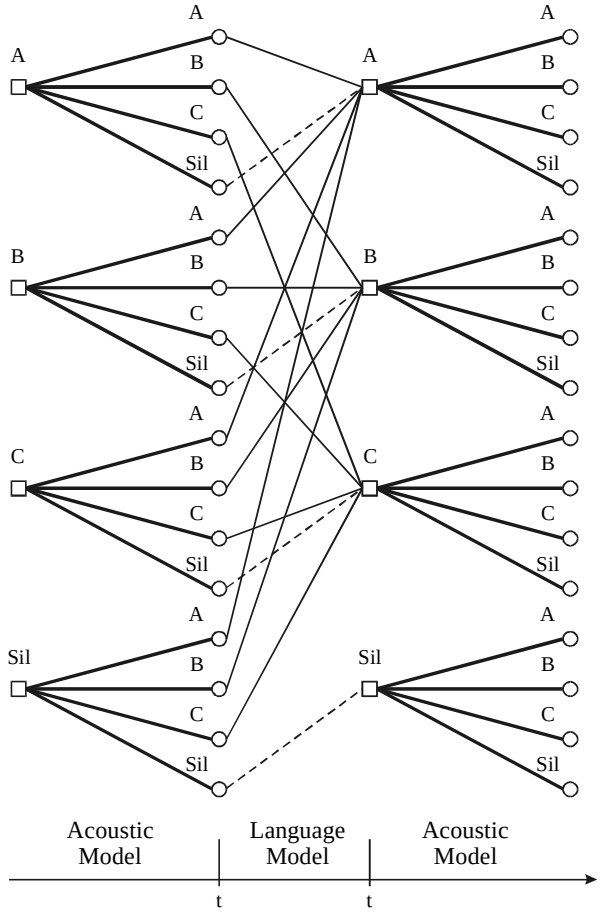
Figure 2. Same lexicon organized in a prefix tree structure.

When organizing a lexicon as a prefix tree, the number of calculations to be performed is reduced when compared to linear organized lexicons, because we don't have to repeat operations already performed. For example, for the words *speak* and *speech*, the two initial phones are the same, so the likelihoods for a given observation will be exactly the same. The task of a tree organized lexicon is to take into account these redundancies and avoid unnecessary calculations.

2.1 INCORPORATING LANGUAGE MODEL .

When using a language model on a tree organized lexicon there are some issues that must be considered. For a bigram language model, for example, the following problem arises: a) the identity of a word *w* under hypothesis is only known after reaching a leaf of the tree; b) the language model probabilities can only be incorporated when the system reaches the terminal state of the second word of the bigram. As a consequence, it's only possible to apply the language model at the end of the tree after taking into account the previous word.

To allow the application of the dynamic programming principles, we must organize the search space in the following manner: for each predecessor word *v*, we introduce a separate copy of the lexical tree so that during the search process we always know the predecessor word *v* when we make the hypothesis of the end of a word *w*. Figure 3 shows this concept for a three word vocabulary *A*, *B* and *C*, where the lexicon tree is shown in a simplified schematic form.



○ word end hypothesis A,B,C
□ root of a tree copy for the history A,B,C
—— : acoustic models recombination within a tree copy
—— : bigram language model recombination
- - - - : word boundary recombination for silence hypothesis

Figure 3. Modification of the search space for the utilization of a bigram language model over a tree organized lexicon.

In the recognition process, in addiction to spoken words, we must account for possible pauses between the spoken words. To handle these pauses, we must have a HMM model for the silence (Sil) and add a separate copy of this model to each tree. Furthermore, for a possible silence at the sentence beginning, we have a separate copy of the lexical tree for the first word of the sentence; this tree has the silence as predecessor. As a result of this approach, the silence model copies do not require a special treatment, and can be processed as regular words of the vocabulary.

There is, however, an exception: at the word boundaries there is no language model for the silence model. As shown in Figure 3, there are two kinds of path extensions and recombinations, namely in the interior of the words or lexical trees and at word boundaries. In the word interior we have the bold lines representing the HMMs transitions. At word boundaries we have the thin and the dashed lines, which represent the bigram language model recombinations. The dashed lines are related to recombinations for intraphrase silence copies.

To start up a new word hypothesis, we must incorporate the bigram probability into the score and determine the best predecessor word. This best score is then propagated into the root of the associated lexical tree, which is represented by the symbol □. The symbol ❍ denotes a word end.

3 BEAM SEARCH PROCEDURE.

The search space size grows up according to the number of words in the system vocabulary. For dictation systems, where vocabulary sizes of tens of thousands words are common, the search space becomes so large that the computational cost becomes prohibitive. However the nonhomogeneous probability distribution over the several paths can be taken into account in order to reduce the computacional complexity. When the number of states is large, at each time a large fraction of them have scores that are very small compared to the maximun likelihood. This fact indicates that the optimum path will probably not pass through these states and we do not have to calculate their scores anymore.

This reasoning lead us to a complexity reduction technique known as *Beam Search* [4], that consists in ignoring, at each time, those states whose accumulated likelihood is less than the maximum likelihood by a given threshold. With this technique, the calculations to expand "bad" nodes are avoided. It´s clear from the nature of this technique that we can lose the best path, but in practice, a convenient choice of the pruning threshold results in a speed gain of an order of magnitude, while introducing a negligible loss in word error rate performance.

4 TESTS.

The tests consisted on a comparative study between two systems, one with linear organized lexicon (One Step) and the other with tree organized lexicon (Herrman-Ney). The index of performance is the recognition time. In fact, it's known that the tree organized lexicon leads to a better performance, but how much better it is and what secrets are inside its implementation are the issues of this work.

Up until the preparation of this article the language model was not implemented yet, and to avoid high word error rates, we decided to make tests in speaker dependent mode only. This limitation however is of no consequence for the test results because we are concerned about recognition times and not about word error rates. It's important to emphasize that either the One Sep or the Herrman-Ney algorithms achieve exactly the same results in terms of word error rate (even the likelihoods assigned to the candidate sentences are exactly the same), i.e., the Herrman-Ney algorithm is just an intelligent way to organize the pronunciation lexicon.

4.1 EXPERIMENTAL SETUP

For the tests we used a locally developed speech recognition system [10][12] based on continuous HMMs and context independent phones as phonetic units.

The configuration used for both systems was: 36 context independent phones as the acoustic sub-units; each sub-unit was modeled by a 3 state HMM with 5 gaussians per mixture per state; feature vectors: Mel , delta-Mel and delta-delta-Mel cepstral coefficients, each one with dimension 12; word duration model and 15 levels for search. The search levels refer to the position of a word in a sentence; for example, the second level corresponds to the second word in the sentence. Then, for a search performed within 15 levels, this means that the sentence can have at most 15 words (including the initial e final silences).

For the recognition times evaluation a Pentium **III** microcomputer with **866** MHz clock and 256 MBytes RAM was used in all the tests reported in this work.

4.2 DATABASE

The sentences were chosen from a work from Alcaim et. al. [1], where 200 phonetically balanced sentences were listed. In these sentences we counted 694 different words.

For the recordings, we used an adult male speaker that spoke all the 200 sentences four times. Three of them were used to train the system and the last one for the tests. These locutions were used for speaker dependent tests.

The recordings were performed in a relatively noiseless room, with a SHURE SM-58 directional microphone, using a SoundBlaster AWE 64 sound card. The sentences were recorded at 11025 kHz sampling rate and 16 bits of resolution. The locutions were manually transcripted, using the Cool Edit 2000 software [12] for viewing the waveform and spectrogram and earphones to listen to them carefully.

4.3 TESTS WITHOUT BEAM SEARCH

In a first set of tests we compared the performance of the two aforementioned systems without any pruning procedure. The results of these initial tests are shown in Table 1.

Table 1. Recognition times for the two search algorithms without any pruning procedure. Mean over 600 locutions.

One Step	01:20 minutes
Herrman-Ney	01:10 minutes

We can observe a little improvement in the recognition times when using a tree organized lexicon.

4.4 TESTS WITH BEAM SEARCH

In a second set of tests the Beam Search pruning procedure was used to eliminate bad paths. With this procedure we obtained the results shown in Table 2.

Table 2. Recognition times using Beam Search with pruning threshold $\lambda = 30$. Mean over 600 locutions.

One Step	42 seconds
Herrman-Ney	16 seconds

4.5 RESULT ANALYSIS

The great time reduction observed is not solely due to the lexicon organized tree and to the pruning threshold, but also to the emission probabilities computation procedure. We exploited some symmetries on the local contributions for the total likelihood in order to reduce the amount of calculations. Next we explain how this was be done.

The recognition of a spoken sentence corresponds to the task of finding the best path in the search space in a minimum likelihood sense. The search space is formed by the HMM models of all words in the vocabulary, repeated at each level. In this work we used a 700 word vocabulary and 15 search levels. So, for a sentence recognition, the system has to process the Viterbi algorithm over 700 x 15 = 10500 word HMM models.

To understand this, let's analyze Figura 4 where we illustrate the Viterbi algorithm working process.

When walking over the trellis, at each new entry, the systems stays in the same state or make a transition to the next state. To decide how the system will behave with this new entry, we calculate a cost function defined as

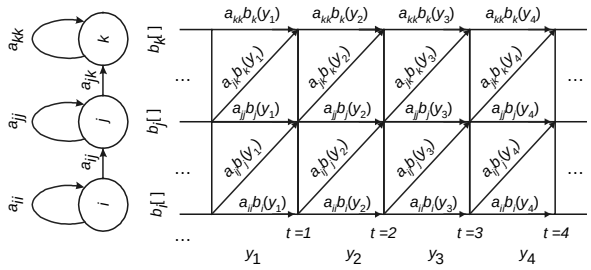


Figura 4. Viterbi algorithm example.

$$v_j(t) = v_i(t-1) + \log(a_{ij}) + \log(b_j(y_t)) \quad \textbf{(1)}$$

where $v_j(t)$ is the accumulated log-likelihood of the best path arriving in state *j* at time *t*, $v_i(t-1)$ is the accumulated log-likelihood of the best path reaching state *i* at time (*t*-1), $\log(a_{ij})$ is the transition log-probability, and $\log(b_j(y_t))$ is the log-probability of emitting symbol y_t in state *j*. This emission log-probability is given by:

$$\log(b_j(y_t)) = \log(\sum_{i=1}^{g} G(\mu_i, \sigma_i^2, y_t) \quad \textbf{(2)}$$

where *g* is the number of gaussians in the mixture, each one with mean μ_i and variance σ_i^2.

As we can see, the computational load to calculate this cost function is very high because we have several logarithms and gaussians to be computed. After multiplying these operations by the number of words, the number of levels and also the number of states in each word, we have an idea of the great effort that must be made in order to recognize a spoken sentence.

There is, however, a symmetry in the search space that can be favorably exploited: when calculating the accumulated log-likelihood using (1), we observe that the local contribution of a transition from state i to the state j, given by

$$\log(a_{ij}) + \log(b_j(y_t)) \quad (3)$$

depends solely of the HMM model being analyzed and the input symbol, and is *independent of the level*. So we don't have to recalculate the local contribution for a given HMM model for each level; it's enough to calculate it for just one level.

In a tree organized lexicon we can achieve an even great amount of economy because the word models are formed by the concatenation of acoustic sub-unit models (context independent phones in our simulations). Because of that, for each input frame (feature vector) we have just to calculate the local contributions of the acoustic sub-units and not the contributions of the whole words.

Another processing economy arises when using the Beam Search procedure over a tree organized lexicon: in a given time, it may happen that one or more acoustic sub-units have all their states pruned in all levels by the Beam Search procedure. So it's not necessary to calculate the local contributions for this sub-unit, because they simply will not be necessary for the search. This is the main reason for the great time reduction observed in Table 2.

5 CONCLUSIONS AND FUTURE WORKS.

In this work we have presented a theoretical study on tree organized lexicons and a comparative study between linear and tree organized lexicons.

In the evaluation tests, the tree organized lexicon system achieved recognition times about 60% smaller than those achieved by the linear organized lexicon system using the Beam Search pruning strategy. This result can be achieved using some symmetries in the search space to avoid unnecessary calculations.

For the future, a bigram language model will be incorporated to the system in order to reduce the word error rate.

Acknowledgements: The authors would like to thank the FAPESP agency for partial funding. Process number 99/01241-2.

6 BIBLIOGRAPHY.

[1] ALCAIM, A., SOLEWICZ, J. A., MORAES, J. A. Frequência de ocorrência dos fones e lista de frases foneticamente balanceadas no português falado no Rio de Janeiro. *Revista da Sociedade Brasileira de Telecomunicações*, 7(1):23-41, Brazil, 1992.

[2] AUBERT,X., DUGAST, C., NEY, H., and STEINBISS, V. Large vocabulary continuous speech recognition on Wall Street Journal data. *In Proceedings of the 1994 International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 129-132. Adelaide. Australia. IEEE. 1994.

[3] AUBERT, X. NEY, H. Large vocabulary continuous speeech recognition using word graphs. *Proceedings of ICASSP*, Detroit, MI, May 1995.

[4] DELLER Jr., J. R., PROAKIS, J. G., HANSEN, J.H.L. *Discrete time processing of speech signals*. MacMillan Publishing Company. New York. 1993.

[5] GUPTA, V.N., LENNIG, M., and MERMELSTEIN, P. Fast search strategy in a large vocabulary word recognizer. *Journal of the Acoustical Society of America*. **84**(6), December 1998.

[6] JELINEK, F. *Statistical methods for speech recognition*. MIT Press. London. 1998.

[7] NEY, H. The use of a one-stage dynamic programming algorithm for connected word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing.* **ASSP-32**(2). April 1984.

[8] NEY, H. & ORTMANNS, S. Dynamic Programming Search for continuous Speech Recognition. IEEE Signal Processing Magazine. September 1999.

[9] RABINER, L. *Fundamentals of speech recognition*. Prentice Hall Press. 1993

[10] YNOGUTI, Carlos Alberto. *Reconhecimento de fala contínua usando modelos ocultos de Markov*. Tese de Doutorado. UNICAMP. Campinas. 1999.

[12] YNOGUTI, C. A. VIOLARO, F. "Um sistema de reconhecimento de fala contínua baseado em modelos de Markov contínuos." *Anais do XVIII Simpósio Brasileiro de Telecomunicações.* 3 a 6 de setembro de 2000. Gramado.

[12] www.syntrillium.com (20/11/2001).