A FAST SEGMENTAL VITERBI ALGORITHM FOR LARGE VOCABULARY RECOGNITION

P. Laface ⋆ *and C. Vair* ⋆ *and L. Fissore* ◇

⋆ Dipartimento di Automatica e Informatica - Politecnico di Torino
Corso Duca degli Abruzzi 24 - I-10129 Torino, Italy E-Mail laface@polito.it
◇ CSELT - Centro Studi e Laboratori Telecomunicazioni
Via G. Reiss Romoli 274 - I-10148 Torino, Italy E-Mail fissore@cselt.stet.it

ABSTRACT

The paper presents a Fast Segmental Viterbi Algorithm. A new search strategy particularly effective for very large vocabulary word recognition. It performs a tree based, time synchronous, left-to-right beam search that develops time-dependent acoustic and phonetic hypotheses. At any given time, it makes active a subword unit associated to an arc of a lexical tree only if that time is likely to be the boundary between the current and the next unit. This new technique, tested with a vocabulary of 188892 directory entries, achieves the same results obtained with Viterbi algorithm, with a 35% speedup. Results are also presented for a 718 word, speaker independent continuous speech recognition task.

1. Introduction

The recognition of very large vocabularies, for automatic directory services through the telephone line, is a challenging research problem. For these applications, the main problems are not only the extremely large perplexity of the task and the similarity of the vocabulary words, but also the search complexity.

There are two main approaches for very large vocabulary recognition. The first one is a two-step strategy of fast lexical access, which first generates a list of acoustically similar words, then rescores these candidate with more accurate models. The second one relies upon phone look-ahead and beam search to reduce the search space. Relevant examples of the latter approach are the works on block Viterbi algorithm [4] and on phoneme look-ahead [3, 6] for isolated word and continuous speech respectively. In this paper, following the second approach, we present a novel Fast Segmental Viterbi Algorithm (FSVA) that uses a time synchronous left-to-right beam search Viterbi algorithm, without any block time delay. The search complexity is reduced not only because we develop time-dependent phonetic hypotheses, and carefully exploit the likelihood of the units previously computed, but mainly because we make active the next subword unit at any given time only if that time is likely to be the location of a boundary between the current and the next unit segments. To decide the "likelihood" of a phonetic unit boundary, we look ahead of the current frame, but rather than using a threshold based look-ahead pruning strategy [3, 6], we make active the next unit only if the time boundary between the considered units reached a "stable" location. Boundaries are detected by means of a very simple and effective process, which rarely misses the exact location of the boundaries, as determined by Viterbi decoding.

Our recognition systems perform acoustic decoding by means of a beam-search Viterbi procedure using a lexical tree to reduce both memory and search costs. During recognition, every arc in the tree is associated with the Markov model of the corresponding subword unit. The procedure scores all word transcriptions corresponding to the terminal nodes that are not pruned after last observation frame has been processed. The lexical tree used for the isolated word experiments merges the 188892 entries that appear in the Italian general telephone directory at least two times and includes 775874 arcs (versus 1628639 units of the linear lexicon) with a potential search space of more than 2 million states.

2. Time-dependent hypotheses

This work has been stimulated by two observations. The first one is based on the analysis of the distribution of the duration of the subword unit hypotheses generated by a classical beam-search Viterbi algorithm (VA) while decoding an utterance. We observed that most of the computational effort is spent expanding tree hypotheses having as their last decoded unit a segment of 3 frames: 3 frames is the minimum duration imposed by the topology of our models.

The vast majority of these segments is, of course, incorrect. Since every wrong segmental hypothesis generates, virtually at every time, other (wrong) hypotheses, the number of short segment hypotheses grows considerably. Although their growth is limited by the Dynamic Programming (DP) recombination and by the

beam-search pruning strategy, the blind activations of units affects the recognition time especially if the lexical tree is very large, or in continuous speech decoding. The second observation is that in a very large lexical tree many arcs are associated to the same phonetic unit. In our tree, for example, the same subword unit appears in 2957 arcs on the average, while the most frequent unit /i@/ - vowel /I/ with silence as its right context - appears 68485 times since most Italian surnames ends by /I/.
From these observations it comes out that a search strategy must try to reduce to a minimum the unit activation times that are likely to be incorrect, and must also use the likelihood of a unit previously computed for every tree arc this unit is associated with. To achieve both goals it is worth formulating the Viterbi algorithm by a notation that explicitly takes into account the boundaries between units rather than considering the detection of these boundaries a useful by-product of the search. This formulation has been introduced in [4] and [6] to exhibit the effect of the boundary between phones and words respectively.
In the rest of the paper we will refer to a branch in the lexical tree including the sequence of arcs a_0, a_1, and a_2 associated with unit u_0, u_1, and u_2 respectively. Moreover, we will refer to "unit u_i" as a shortcut for "the instance of units u_i associated with arcs a_i".
Let $o_1 \dots o_T$ be a sequence of observation frames and let u belong to a set of U models, time τ be the time boundary between unit u_i and u_{i+1}, or more precisely, the start time of unit u_{i+1}, and let's also define:

- $p^t_\tau(s, u)$, the log probability (likelihood) that the best sequence of states $s_1 \dots s$ of unit u, with $1 \leq s \leq S(u)$, and $S(u)$ the final state of unit u, produces the observation frames $o_\tau \dots o_t$.
- $h^t_\tau(u) = p^t_\tau(S(u), u)$, the log probability that unit u produces the observation frames $o_\tau \dots o_t$
- $H_t(a_i)$, the log probability of the sequence of arcs that generated $o_1 \dots o_t$, with a_i as last arc of the sequence.

Our goal is a fast technique for computing the log probability $H_T(a^\star)$ for every tree arc $a^\star$ that ends in a terminal node. Using the above introduced definitions, the auxiliary function:

$$K_t(\tau, a_1, a_2) = H_{\tau-1}(a_1) + h^t_\tau(u_2) \qquad (1)$$

represents the log probability, as a function of time τ, of the best sequence of arcs generating $o_1 \dots o_t$, with a_2 as last arc of the sequence, and with τ as boundary between unit u_1 and u_2.
$H_t(a_2)$ can be computed recursively, for $t > 1$, t being the time of u_2 completion, by:

$$H_t(a_2) = \max_{\tau \leq t} K_t(\tau, a_1, a_2) \qquad (2)$$

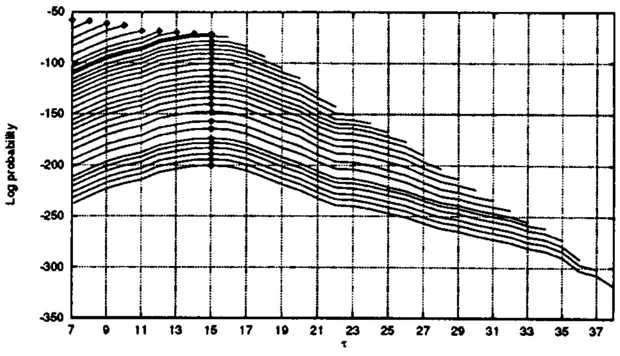

Figure 1: $K_t(\tau,$/fe/,/eR/$)$ as a function of τ, the time boundary between unit /fe/ and /eR/ in an utterance of word /FERRARI/.

and the best boundary between unit u_1 and u_2 by:

$$\tau_t(a_1, a_2) = \operatorname*{argmax}_{\tau \leq t} K_t(\tau, a_1, a_2) \qquad (3)$$

This relation allows the correct boundary to be detect, but, unfortunately, is not useful in this form, since it is expensive to find the time boundary $\tau_t(a_1, a_2)$ that optimizes $H_t(a_2)$, for every path originating at the tree root that is still alive at time t. To reduce the complexity of this optimization it is interesting to analyze the behavior of $K_t(\tau, a_1, a_2)$, as a function of τ, after setting to a fixed value the startup time of the previous unit a_1; this stable boundary is referred to in the following as $\tau_0(a_0, a_1)$. Fig. 1 shows the set of functions $K_t(\tau, a_1, a_2)$, as a function of τ - $\tau_0(a_0, a_1) < \tau \leq t$ -. Recall that $K_t(\tau, a_1, a_2)$ gives the value of $H_t(a_2)$ at time t if the boundary between units a_1 and a_2 is τ. The unit a_1 and a_2 presented in this figure are /fe/ and /eR/, the initial units in the transcription of word /FERRARI/, and the startup time of unit /fe/ is fixed at frame 4; there is a function for each completion time t of unit a_2 ($9 \leq t \leq 40$), and a diamond marks the coordinates of the point where each function reaches its maximum value.
It is easy observing that this set of curves follows a typical parallel pattern and that the maximum of each function, corresponding to the best boundary between the units, is stable for a wide range of the completion time t of unit /eR/. This is the typical behaviour of these functions for every pair of units, even if we have found a few counterexamples.
Another interesting feature of these functions is that the diamond marker, corresponding to the maximum of a function, before reaching a stable position, is located at the edge of a curve $K_t(\tau, a_1, a_2)$. This edge point identifies the boundary between unit u_1 and u_2 corresponding to the minimum duration of unit u_2. This behavior is not surprising because, if the boundary $\tau_t(a_0, a_1)$ between the previous units is correct, as long as t has not yet reached the true completion time

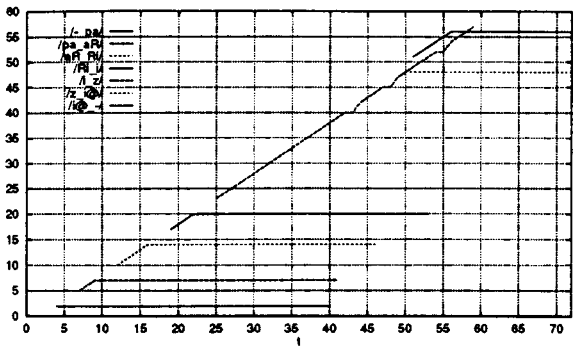

Figure 2: Unit boundaries $\tau_t(a_i, a_{i+1})$ as a function of time t in an utterance of word /PARISI/.

of unit u_1, the observed frames are better recognized by the model of unit u_1 rather than by the model of unit u_2. As a consequence, when it is not stable, a unit boundary moves along a line parallel to the diagonal in a plane whose x and y axes are the completion time t of unit a_2 and $\tau_t(a_1, a_2)$, respectively. An example of this plane, for an utterance of word /PARISI/, is shown in Fig. 2.

3. Fast Segmental Viterbi Algorithm

To cope with the problems observed in the previous Section, we propose a new fast search technique. Rather than looking ahead for fast matching or for obtaining a better pruning threshold as in [3], our approach expands the current path hypothesis ending in arc a_i, with the next arc a_{i+1}, only if the boundary between the associated units u_i and u_{i+1} is likely to be located at the current time frame.

Suppose that we have observed the sequence of input frames $o_1 \dots o_{t-1}$, and that the likelihood of the hypotheses ending in arc a_1 at time $t-2$, $t-1$, and t have been computed.

The basic idea of the FSVA strategy is to avoid activating next unit a_2 at time t if the time boundary $\tau_{t+mindur(u_2)}(a_1, a_2)$ is not stable. Since it is stable only if a local maximum of function $K_t(\tau, a_1, a_2)$ is located in $t \equiv \tau_{t+mindur(u_2)}(a_1, a_2)$, rather than performing the expensive search of equation (2), we activate next unit u_2 of arc a_2, at current time t, only if:

$$\operatorname*{argmax}_{t-1<\tau<t+1} K_{t+mindur(u_2)}(\tau, a_1, a_2) = t \qquad (4)$$

where $mindur(u_2)$ is the minimum number of input frames that allow the Markov states of unit u_2 to be traversed; $mindur = 3$ in our case for every unit, excluding the single state silence model.

If condition (4) is true, the segmentation boundary $\tau_{t+mindur(u_2)}(a_1, a_2)$ is stable at time t at least for two time frames. Since $\tau_{t+mindur(u_2)}(a_1, a_2) \equiv t$ is a potential correct boundary, we allow the activation of next unit at the current time performing the DP update of the likelihood $H_{t+mindur(u_2)-1}(a_2)$ according to the following equation:

$$H_{t+mindur(u_2)-1}(a_2) = \max \begin{cases} H_{t+mindur(u_2)-1}(a_2) \\ H_{t-1}(a_1) \ + \\ \quad h_t^{t+mindur(u_2)-1}(u_2) \end{cases}$$

Since next unit is being activated at current time t, and it is completed at time $t + mindur(u_2) - 1$, the boundary $\tau_{t+mindur(u_2)-1}(a_1, a_2)$ is set to t.

DP updating of likelihood $H_t(a_2)$ is also performed at current time t, using the following equation:

$$H_t(a_2) = \max(H_t(a_2), H_{\tau_p}(a_1) + h^t_{\tau_p}(u_2)) \qquad (5)$$

where $\tau_p = \tau_{t-1}(a_1, a_2)$ is the start time of unit u_2 currently associated with hypothesis $H_t(a_2)$.

Equation (5) can be rewritten as:

$$H_t(a_2) = \max(H_t(a_2), H_{t-1}(a_2) - h^{t-1}_{\tau_p}(u_2) + h^t_{\tau_p}(u_2)) \qquad (6)$$

we can use, thus, a time synchronous, left-to-right beam search Viterbi algorithm to perform the recursion since the needed information is related to the previous and current frame only.

3.1. Consistency of FSVA

The strategy is not consistent since it can fail to detect the correct boundary $\tau_0(a_1, a_2)$ located in t, for two reasons:

- $\tau_t(a_1, a_2)$ is not stable
- condition (4) is true, but the maximum of function $H_{t+mindur(u_2)}(a_2)$ in (6) is obtained for the second argument.

In the example shown in Fig. 2, our algorithm fails to find a correct boundary located at frame 38 since the boundary is not stable there. It finds instead an incorrect boundary at frame 45, where a short "stable" location exists. The analysis of the behavior of functions $H_t(a_2)$ and $\tau_t(a_1, a_2)$, presented in the Section 2., however, shows that probability of missing the correct boundary is very low. To ensure that the boundaries derived by FSVA and by Viterbi algorithm (VA) agree, we performed a forced segmentation on each test utterance by means of VA. Comparing the 96933 "correct" boundaries with the ones generated by FVSA we found that they perfectly agree 98.1% of the time. Moreover, we found that almost always these errors affect a pair of adjacent segments of the same word, and that the deviation from the correct boundary is just of one frame for 1128 of the 1782 incorrect boundaries.

	Viterbi	FSVA		
Beam threshold	30	30	35	40
Inclusion rate (%)	87.8	86.7	88.0	88.8
Avg unit startup per frame	6360	3913	5632	7496
Avg CPU time (sec)	3.7	1.7	2.4	3.8

Table 1: Recognition results with a 188892 word vocabulary

4. Experimental results

The Isolated Word recognition system that has been used as benchmark is based on a total of 203 Discrete Density Hidden Markov Models of subword units. The test database includes a total of 12720 utterances, collected through a PABX, of a set of 600 surnames, pronounced by 120 speakers. The surnames in this database were selected from the 188892 entries both for their occurrence frequency, and also to assure a large phonetic coverage. Table 1 presents the results of the recognition experiments performed on a DEC AXP 3000/500 workstation - the percentage of inclusion rate refers to the top 50 candidates -. With respect to the VA, a 54% speedup is obtained at the cost of a slightly increase of the error rate using the same beam search pruning threshold, while the same inclusion rate is achieved saving 36% of the CPU time using a larger threshold, or even better results are achieved using the same average CPU time.

In the second set of experiments we tested the performance of the FSVA with vocabularies of different size. For a "small" 600 word vocabulary, the fixed overhead due to the evaluation of time-dependent likelihoods and the added complexity of the strategy increase the search costs of FSVA with respect to VA, but for a vocabulary size of more than 5000 words FSVA is faster, other results can be found in [5].

We compared also our search based on the lexical tree with a strategy that has some similarity with the ones proposed in [1, 4]. The search is performed, in a first step, on a graph which merges units that appear in the same position within a word transcription (thus overgenerating the vocabulary). The output of this first step is a lattice of unit hypotheses which are rescored by means of a very fast backward A^* lexical tree search. We used graphs of m-gram units (with $m = 1 \ldots 3$). The results of these experiments show that a 1-gram graph does not give a good lexical representation since it factorizes too much the lexicon and doubles the number of errors. For bigram and trigram units we observed a very small reduction of the search space (13% and 1% respectively), but this reduction does not correspond to a reduction of CPU time, which on the contrary doubles with respect to the tree based approach. The reason for this behavior is that many wrong hypotheses are generated by the approximate lexical representations that cannot be traded for the very small reduction of the search space.

Finally, preliminary experiments have been performed with a continuous speech system based on a total of 310 Continuous Density HMM of subword units, with 15 Gaussian mixtures per state, and without any language model. It has been evaluated on 600 sentences collected from 10 speakers in a controlled environment, these sentences refer to a train timetable inquiry task, with a vocabulary of 718 words [2]. In this case the evaluation compares the complexity of the search for the same performance in terms of word accuracy: the average number of active units per 10 ms frames is more than doubled for the Viterbi Algorithm with respect to FSVA using the same beam search threshold. However, the reduction of search cost for the FSVA is 10% only due to the "small vocabulary" and to the lack of language model. We are presently applying FSVA to continuous speech with bigram constraints, a task in which the multiplication of word copies and the much larger search space should better demonstrate the effectiveness of this strategy.

5. Conclusions

A fast search strategy for large vocabulary recognition has been presented that, using larger beam search thresholds, is able to achieve, in less time, the same or better results with respect to a Viterbi algorithm. It is worth noting that the look-ahead strategy of [3] can be directly included in our framework in conjunction with our condition (4).

6. References

[1] J.K. Chen, F. Soong, and L.S. Lee, "Large Vocabulary Word recognition based on Tree-Trellis Search", ICASSP-94, pp. II-137-140, 1994.

[2] L. Fissore, E. Giachin, P. Laface, and P. Massafra. "Using Grammars in Forward and Backward Search", EUROSPEECH 93, pp. 1525-1528, Berlin, 1993.

[3] R. Haeb-Umbach, H. Ney, "A Look-Ahead Search Technique for Large Vocabulary Continuous Speech Recognition", EUROSPEECH 91, pp. 495-498, Genova, 1991.

[4] P. Kenny, et al., "A^*-Admissible Heuristics for Rapid Lexical Access", IEEE Transactions on Speech and Audio Processing, Vol. 1, n. 1, pp. 49-58, 1993.

[5] P. Laface, M. Nicolazzo, L. Fissore, "Word Recognition for Very Large Vocabularies", CRIM/FORWISS Workshop on Progress and Prospects of Speech Research and Technology, pp. 93-101, Munich, Germany, 1994.

[6] H. Ney, "Architecture and Search Strategies for Large-Vocabulary Continuous- Speech Recognition", Proc. NATO ASI Bubion, Spain June 1993, pp. 59-84.