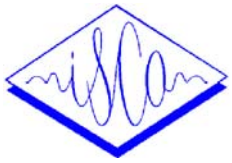

6th European Conference on
Speech Communication and Technology
(EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

OPTIMIZATION OF DYNAMIC REGIMES IN A STATISTICAL HIDDEN DYNAMIC MODEL FOR CONVERSATIONAL SPEECH RECOGNITION

Jeff Ma and Li Deng

Department of Electrical and Computer Engineering,
University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

ABSTRACT

This paper reports our on-going work aimimg to improve the performance of a novel speech recognizer based on an underlying statistical hidden dynamic model of phonetic reduction in the production of conversational speech. We have developed a *path-stack* search algorithm which efficiently computes the likelihood of any observation utterance while optimizing the dynamic regimes in the speech model. The effectiveness of the algorithm is tested in simulation experiments It is also tested on Switchboard data where the optimized dynamic regimes by the search algorithm are compared with those from exhaustive search. Finally, we show speech recognition results on Switchboard data that demonstrate improvements of the recognizer's performance compared with use of the dynamic regimes heuristically set from the phone segmentation by a state-of-the-art HMM system.

1. INTRODUCTION

This paper reports our recent work on automatic optimization of dynamic regimes in a statistical coarticulatory model for the hidden vocal-tract-resonance (VTR) dynamics, and reports on the resulting improved performance in spontaneous speech recognition on the Switchboard corpus. This constitutes an early phase of the follow-on work supported by NSF following the six-week 1998 NSF/DoD Workshop on Language Engineering carried out at Johns Hopkins University.

In the earlier work detailed in [4] and reported therein, we found that a most important factor affecting the our new recognizer's performance is the boundaries of dynamic regimes in the dynamic VTR model of speech. When the boundaries were manually (via hours and hours of spectrogram reading) adjusted from those automatically (often with gross errors) determined from an HMM system, drastic reduction of word error rate in the Switchboard task was consistently observed. This motivates the current work aiming to develop an efficient (segmentation) algorithm which can automatically determine the optimal dynamic regimes in training the recognizer and in scoring spontaneous speech utterances. The optimality criterion, chosen as the likelihood on the observation data (MFCC sequences), is based on the dynamic VTR model used to represent the dynamic patterns of speech, rather than based on other inconsistent forms of model such as HMM. This achieves the desirable goal of consistency modeling in the entire system.

2. THE VTR-BASED DYNAMIC MODEL

In this section, we provide a brief overview of the new dynamic model as the context in which the new optimization algorithm has been developed. Details of the model and some early experiments are provided in [4] and [6]. This speech model, based on VTR dynamics, has been formulated in mathematical terms as a constrained and simplified nonlinear dynamic system, and is a special version of the general statistical hidden dynamic model described in [3, 2].

Briefly, we have the following linear state equation on the dynamic, target-directed behavior for the VTR state variables $Z(k)$:

$$Z(k+1) = \Phi Z(k) + (I - \Phi)T + W(k), \qquad (1)$$

where T and Φ are the VTR target and "time constant", respectively, both distinct for each phone in a speech utterance (phone index omited for clarity).

The observation equation in the dynamic system model is described by

$$O(k) = h[Z(k)] + V(k), \qquad (2)$$

where the acoustic observation $O(k)$ is MFCC measurements computed from a conventional speech preprocessor, $V(k)$ is the additive observation noise modeled by an i.i.d., zero-mean, Gaussian process with covariance matrix R, intended to capture residual errors in the mapping from $Z(k)$ to $O(k)$. The multivariate nonlinear mapping, $h[Z]$, is implemented by a global multi-layer perceptron (MLP).

This new speech model underlying our recognizer is based on new principles of modeling hidden dynamics of speech production in the VTR domain. This model consists of two separate components which accommodate separate sources of speech variabilities.

3. THE PATH-STACK ALGORITHM FOR OPTIMIZING DYNAMIC REGIMES

We first explain why the standard dynamic programming approach (Viterbi algorithm) can *not* be applied directly to search for optimal dynamic regimes in the new dynamic speech model, no matter how small the number of the dynamic regimes is. In short, the difficulty arises because of the continuity constraint imposed across dynamic regimes in the hidden variable domain. A consequence of this constraint is this: at any fixed node in the trellis diagram, the local scores into a fixed future node will in general be different depending on the past history arriving at the current

node[1]. We first give an example to illustrate this point. We then describe a reasonably effective algorithm we have developed to overcome such a difficulty and to provide an approximate solution.

3.1. Example

In the trellis depicted in Fig.(1), we show three left-to-right dynamic regimes (S_1, S_2, and S_3) for simplicity reasons. At time frame $t = 3$, there are two possible paths entering regime S_2: "$b11 \rightarrow b22 \rightarrow b23$", and "$b11 \rightarrow b12 \rightarrow b23$". Note that at time $t = 2$ these two paths have an identical value, $\hat{Z}(1|1)$, $P(1|1)$, to initialize the Extended Kalman Filter (EKF). But since the two paths use distinct model parameters (T, Φ), one from S_1 and the other from S_2, they generate different likelihood scores, $L_1(t = 2)$ and $L_2(t = 2)$, as well as different filtered values, $\hat{Z}_1(2|2)$, $P_1(2|2)$ and $\hat{Z}_2(2|2)$, $P_2(2|2)$. At frame $t = 3$, these two paths will use the same model (S_2) parameters for the EKF (in likelihood computation) when they enter into regime S_2, but their initial points, $\hat{Z}_1(2|2)$, $P_1(2|2)$ and $\hat{Z}_2(2|2)$, $P_2(2|2)$, for the EKF are different. Therefore, these two paths will generate distinct regime-bound, local scores, $L_1(t = 3)$ and $L_2(t = 3)$, and distinct filtered values $\hat{Z}_1(3|3)$, $P_1(3|3)$ and $\hat{Z}_2(3|3)$, $P_2(3|3)$, at frame $t = 3$ and for regime (S_2).

In Viterbi algorithm for the conventional HMM, at this point the path with lower likelihood of among the two (i.e., the smaller one of $L_1(t = 1) + L_1(t = 2) + L_1(t = 3)$ and $L_2(t = 1) + L_2(t = 2) + L_2(t = 3)$) can be dropped for future path growth without losing global optimality. This is so because the two paths would give an identical score when entering node "b34" (or "b24") and would behave identically in the future path expansion.[2]

However, if we were to drop the low-score path at node "b23" as in the Viterbi algorithm for HMM, we would lose global optimality. This is so because both paths emanating from "b23" will generate different local scores while entering node "b34" (due to different initial values, $\hat{Z}_1(3|3)$ and $\hat{Z}_2(3|3)$, for EKF) and would hence behave differently in the future path growth.

The problem exemplified above associated with the dynamic model and with use of EKF applies to almost all nodes in the trellis. That is, due to the continuity constraint in the VTR domain from one dynamic regime to the next,[3] no path drop is allowing without losing global optimality. In theory, the search space expands exponentially with time.

3.2. The Path-Stack Algorithm

We have developed an approximate solution to the above problem of exponentially growing computation with controlled loss of optimality while gaining reasonable computational efficiency. The essence of this algorithm is to use a path-stack at each node in the trellis to maintain a sufficient but limited number of the promising paths.

[1] It is the difficulty of such a type which prevents IBM from further developing a model that is also substantially different from the conventional HMM [1]. Note also that the computational complexity associated with most versions of stochastic segment models [5] arises from somewhat different causes since generally no explicit constraints across segments are imposed.

[2] This is the essence of dynamic programming or Viterbi algorithm.

[3] This is implemented by forcing the end value of the earlier regime to initialize the EKF for the next regime.

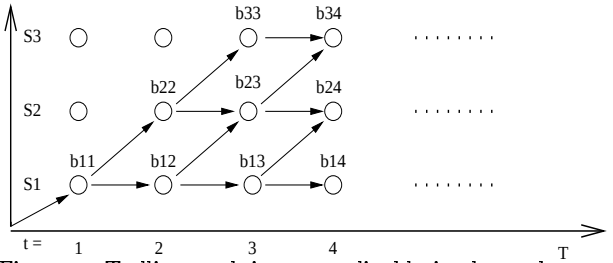

Figure 1: Trellis search is *not* applicable in theory because local scores into a future node depend on the history into the current node

While developing this "path-stack" algorithm, we found that the initial filter error covariance, $P(k|k)$, is of much less importance in determining likelihood scores than the initial filter value, $\hat{Z}(k|k)$. It was found that if the filtered values of two paths were close to each other then the behavior of these two paths in the future would be very similar. In the example of Fig.(1), when the filtered values, $\hat{Z}_1(3|3)$ and $\hat{Z}_2(3|3)$ (corresponding to the two paths at node "b23" at frame $t = 3$), are not far apart, then they will produce similar likelihood scores (using the same model parameters of S_3) when they enter node "b34" at time $t = 4$. Further, due to the asymptotic, target-directed property of the VTR dynamics modeled by Eqn.(1), the filtered values of the two paths at time $t = 4$, $\hat{Z}_1(4|4)$ and $\hat{Z}_2(4|4)$, will become more similar in value. This will guarantee the similarity of scores at future time frames associated with these distinct two paths occurring at an earlier time. Therefore, if we drop one (with a lower score) of these paths with similar filtered values at an early time, the loss of global optimality would be minimal.

In order to prevent the number of paths from increasing exponentially with time, we add a path stack at each node and time and limit the size of it to a fixed value.[4] The size of the path-stack defines how many paths are kept for the node. If the size is small, the path dropping becomes heavy. By adjusting the stack size, we strike a balance between the degrees of optimization and of search efficiency.

Based on the above observation and computational constraint, we have developed the following operation on the path-stacks: At time $t = k$ and node b, calculate the path likelihood of the n-th path ending at node b, $L(b, n, t = k)$, and calculate the filtered (by EKF) values, $\hat{Z}_n(k|k)$ and $P_n(k|k)$. Then, compute the distance, D_{ni}, between $\hat{Z}_n(k|k)$ and $\hat{Z}_i(k|k)$ of the i-th path already in the path-stack of node b. Choose the path whose filter value is closest to $\hat{Z}_n(k|k)$ (say j-th path). If the path-stack of node b is not full and the distance D_{nj} is greater than a preset threshold, insert the new path into the path-stack. Otherwise, if the new path has higher path score, substitute it for the early stored j-th path, if not, drop the new path.

[4] Investigation of using an adaptive stack size is currently underway.

The entirety of the path-stack algorithm as we have implemented in this work for optimizing dynamic regimes and for (approximate) maximum likelihood scoring is shown in Fig.(2). T is the utterance length, N is the total number of nodes in a lattice or in an N-best list, and MAXpath is the maximum number of paths allowed to be kept in a path stack.

The computation complexity of this algorithm is proportional to K*T, where K is the size of the path-stack (pre-determined) that need to be processed at each node in the search trellis. This algorithm turns the original search problem which is exponential in T (tree search) into one which is only linear in T.

```
For each frame , O(t), in the observation squence of an utterance (0 < t < T+1)
    best_score(t) = negative_infinity;
    For each node (phone)  i  in the lattice (or in a N-best list) of the utterance  (0 < i <N+1)
       Let  n_path_in_stack_of_node_i  =  0;
       For each node  j  (not pruned) which can enter node i  (0 < j < N+1)
          For each path  k existing in the path stack of node j at time  t-1
             Calculate acoustic score  L1(O(t)| i, k),  hidden dynamics  Z(k,t) and it's error covariance  P(k,t);
             Let  min_distance  =  infinity  and  min_path_index = 0;
             For each path  l  existing in the path stack of node  i  at time  t
                Let  distance = | Z(i, l, t) - Z(k,t) | ;
                if  ( distance  <  min_distance )
                   min_distance  =  distance;
                   min_path_index  =  l;
                endif
             End
             Let  path_likelihood = L(j, k, t-1)  +  L1(O(t)| i, k);
             if  ( (n_path_in_stack_of_node_i  <  MAXpath )  and  (min_distance  >  Thres_deletion) )
                L( i, n_path_in_stack_of_node_i, t)  =  path_likelihood;
                Z( i, n_path_in_stack_of_node_i, t ) =  Z( k, t );
                P( i, n_path_in_stack_of_node_i, t ) =  P( k, t );
                n_path_in_stack_of_node_i  ++;
                Remeber the back pointer from  ( i,  n_path_in_stack_of_node_i ) to  ( j, k );
             else if
                L ( i, min_path_index, t )  =  path_likelihood;
                Z ( i, min_path_index, t )  =  Z ( k, t );
                P ( i, min_path_index, t )  =  P ( k, t );
             else
                Drop the new path and keep the stack untouched;
             endif
             if  ( path_likelihood  >  best_score (t)  )
                Let  best_score (t)  =  path_likelihood;
             endif
          End
       End
    End
    For each  node  i  in the lattice and each path k in the path stack of node  i
       if  ( best_score (t)  -  L( i, k, t )  >  Thres_pruning )
          Prune  path  ( i, k)  from the search space;
       endif
    End
End
Choose the path with maximum likelihoood.  maxpathlikelihood = max L( i, k, T) over node i  and path k in the stacks.
Trace  back to obtain the segmentation of dynamics and it's associated speech units.
```

Figure 2: The Path-Stack Search Algorithm

4. SIMULATION EXPERIMENTS

The path-stack algorithm has been tested first on simulated data. We choose five different models (regimes) to generate simulation data. Their system parameters Φ and T are as follows (Φ is diagonal matrix):
Model-a: $diag(\Phi) = [0.4, 0.4, 0.4], T = [0.18, 1.5, 2.2]$;
Model-b: $diag(\Phi) = [0.5, 0.5, 0.5], T = [0.42, 2.29, 3.21]$;
Model-c: $diag(\Phi) = [0.3, 0.3, 0.3], T = [0.60, 1.4, 2.1]$;
Model-d: $diag(\Phi) = [0.7, 0.4, 0.45], T = [0.3, 1.9, 2.7]$;
Model-e: $diag(\Phi) = [0.7, 0.6, 0.3], T = [0.6, 2.4, 3.5]$.
All the models have the same measurement equation. Each model generates six samples with both Gaussian state noise and measurement noise added. By concatenating them, we have 30 simulated observation samples. From these samples we search for the dynamic regime boundaries using the path-stack algorithm, and then compare the results with the true regime boundaries.

We use a fixed MLP for the nonlinearity $h(\cdot)$ and fixed stack size of five in the simulation. The results with two different levels of noises are shown in Fig.(3) and (4), respectively. With a smaller noise level (Fig.(3)), our path-stack algorithm outputs identical dynamic regions to the true ones. With the noise level increased by ten folds (Fig.(4)), the algorithm commits three frames of errors (marked by "$\hat{}$"). This is caused by the nonrobustness of the Kalman filter to large noises.

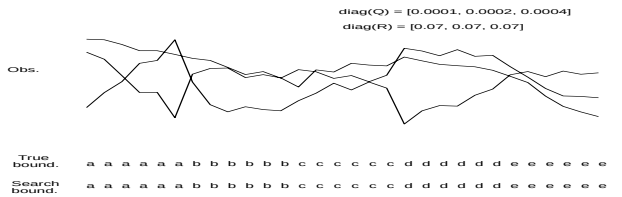

Figure 3: optimized regimes for system with low noise level

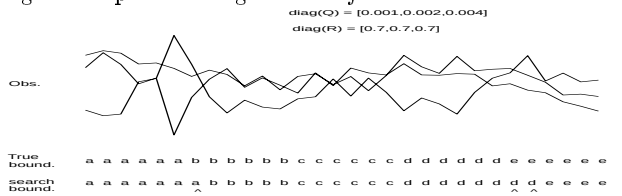

Figure 4: optimized regimes for system with high noise level

5. TESTING THE ALGORITHM ON SPEECH

Built on the reasonable success of the algorithm on the simulation data shown above, we now test the algorithm on Switchboard data.

We first compare the results of the algorithm with exhaustive search. Because it is only possible to use a short utterance to do exhaustive search, we choose an utterance with three phones ("b", "uh" and "t") with 12 frames in total. The exhaustive search results on the boundaries of the dynamic regimes (one for each phone) are listed in Table (1). The search results by the path-stack algorithm, listed as utterance likelihood scores and regime durations as a function of the path-stack size are listed under the column "Utt1" of Table(2). The results show that with the stack size of one, the algorithm fails to pick up the optimal path (determined by the exhaustive search) the algorithm succeeds with the stack size increased to two.

We have tested the algorithm for many Switchboard utterances of much longer lengths; two of them are shown also in Table (2). Stack sizes of three and four are sufficient to produce optimal regime boundaries for both utterances,

"b"	"ah"	"t"	log-L	"b"	"ah"	"t"	log-L
1	1	10	-71196.4	4	2	6	-71301.1
1	2	9	-71160.4	4	3	5	-71297.2
1	3	8	-71077.5	4	4	4	-71339.3
1	4	7	-71049	4	5	3	-71402.9
1	5	6	-71008.6	4	6	2	-71630.6
1	6	5	-71004.6**	4	7	1	-71593.5
1	7	4	-71046.8	5	1	6	-71260
1	8	3	-71110.4	5	2	5	-71255.5
1	9	2	-71338	5	3	4	-71297.6
1	10	1	-71301	5	4	3	-71361.2
2	1	9	-71323.6	5	5	2	-71588.9
2	2	8	-71245	5	6	1	-71551.8
2	3	7	-71217.5	6	1	5	-71284.2
2	4	6	-71177.5	6	2	4	-71326.1
2	5	5	-71173.6	6	3	3	-71389.7
2	6	4	-71215.8	6	4	2	-71617.3
2	7	3	-71279.4	6	5	1	-71551.8
2	8	2	-71507	7	1	4	-71304.6
2	9	1	-71470	7	2	3	-71367.7
3	1	8	-71330.6	7	3	2	-71595.2
3	2	7	-71303.2	7	4	1	-71558.2
3	3	6	-71263.3	8	1	3	-71363.7
3	4	5	-71259.4	8	2	2	-71586.3
3	5	4	-71301.5	8	3	1	-71550.4
3	6	3	-71365.1	9	1	2	-71432.2
3	7	2	-71592.8	9	2	1	-71425.7
3	8	1	-71555.7	10	1	1	-71328.8
4	1	7	-71342.9				

Table 1: Exhaustive search for Utt1 in Table 1

stack size	Utt1(frms-in-phn) 12 frms,3 phs	Utterance2 140frms,18phs	Utterance3 194frms,22phns
1	-71110.4 (1 8 3)	-33592.5	-96535.9
2	-71004.6 (1 6 5)	-33592.5	-96535.9
3	-71004.6 (1 6 5)	-33577.0	-96454.1
5	-71004.6 (1 6 5)	-33577.0	-96432.4
10	-71004.6 (1 6 5)	-33577.0	-96432.4
50	-71004.6 (1 6 5)	-33577.0	-96432.4

Table 2: Search results with different path stack size

as evidenced by the converging likelihood scores. While testing the algorithm for many randomly selected utterances, we find in general that a path-stack size of five is sufficient for the success of the algorithm. This value is hence used in speech recognition experiments which we report below.

6. SPEECH RECOGNITION EXPERIMENTS

One male speaker's data (speaker ID: 1028) was extracted from "train-ws97-dev-l" training set of Switchboard and used for the training of the dynamic models. Such data consist of 966 training utterances. One single MLP was trained for the entire recognizer, common for all phones. During the recognizer testing, N-best rescoring paradigm was used to compare the new VTR-based recognizer, with the dynamic regimes optimized using the path-stack algorithm, with the earlier version of the VTR-based recognizer using fixed dynamic regimes from the HMM phone segmentation [4]. The test set consists of a total of 23 male speakers comprising 24 conversation sides, 1241 utterances, and 50 minutes of speech. In the new recognizer, given the optimized dynamic regimes for each phone of each hypothesis in the N-best list, the computation of the acoustic likelihoods is performed efficiently by running the EKF algorithm for each hypothesis in the N-best lists. At the time of this writting, we have obtained the results for N=5 in the N-best evaluation experiments. The N-best lists are produced by a high-quality triphone-based HMM system.

Two VTR Systems	Ref+5	5-Best
Fixed Dynamic Regimes (by HMM)	29.4%	54.0 %
Optimized Dynamic Regimes	27.2%	53.8 %

Table 3: Test results on Switchboard (1241 utterances)

Table (3) lists the word error rates, with (Ref+5) and without (5-best) adding the reference transcriptions to the N-best (N=5) lists, for two VTR-based systems. One system uses fixed dynamic regimes (benchmark) and the other optimized ones. Some reasonable error rate reduction (close to 10% for the Ref+5 case) has been achieved by using the preliminary implementation of the path-stack algorithm described in Section 3 with a very small stack size.

7. CONCLUSION AND FUTURE WORK

We have developed a "path-stack" algorithm for optimizing the dynamic regimes in a statistical hidden dynamic model which underlies our new recognizer designed for conversational speech exhibiting strong phonetic reductions. The evaluation results have not been as striking as we had hoped for based on the use of hand segmented dynamic regimes (by spectrogram reading). In our future work, we will refine the algorithm and use dynamically-adapted stack sizes. We are currently also investigating other possibilities for joint regime optimization and likelihood computation using the methods developed for dynamic systems with switching parameters in the fileds of time series analysis, neural networks, econometrics, and automatic control.

Acknowledgments:
This work has been supported by the National Science Foundation, U.S. (via Johns Hopkins University) and by Natural Science and Engineering Research Council of Canada. We thank Prof. Fred Jelinek for the encouragement, discussions, and support of this work.

8. REFERENCES

[1] Bakis R. "Coarticulation modeling with continuous-state HMMs," *Proc. IEEE Workshop Automatic Speech Recognition*, Arden House, N.Y., 1991, pp. 20-21.

[2] Deng L. "A dynamic, feature-based approach to the interface between phonology and phonetics for speech modeling and recognition," *Speech Communication*, Vol. 24, No. 4, pp. 299-323, 1998.

[3] Deng L. "Computational models for speech production," in *Computational Models of Speech Pattern Processing* (NATO ASI), 1997.

[4] Deng L. and Ma J. "A statistical coarticulatory model for the hidden vocal-tract-resonance dynamics," *these proceedings.*

[5] Ostendorf M. "From HMMs to segment models: A unified view of stochastic modeling for speech recognition" *IEEE Trans. Speech Audio Proc.*, Vol. 4, 1996, pp. 360-378.

[6] Picone J., S. Pike, R. Reagan, T. Kamm, J. Bridle, L. Deng, J. Ma, H. Richards, M. Schuster. "Initial evaluation of hidden dynamic models on conversational speech," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, March 1999, pp 109-112.