A path-stack algorithm for optimizing dynamic regimes in a statistical hidden dynamic model of speech

Jeff Z. Ma† and Li Deng‡

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

Abstract

In this paper we report our recent research whose goal is 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 on the speech data in the Switchboard corpus, in which the optimized dynamic regimes computed from the algorithm are compared with those from exhaustive search. We also present speech recognition results on the Switchboard corpus that demonstrate improvements of the recognizer’s performance compared with the use of the dynamic regimes heuristically set from the phone segmentation by a state-of-the-art hidden Markov model (HMM) system.

1. Introduction

In recent years, weaknesses of the hidden Markov model (HMM) as a speech model for use in speech recognition have been carefully explored by a number of research groups, from which several types of generalized models (commonly called stochastic segment models in the literature) have been developed as an alternative underlying model in speech recognizers (cf. Deng, 1992; Ostendorf, Digilakis & Kimble, 1996; Siu, Iyer, Gish & Quillen, 1998; Holmes & Russell, 1999). The main weaknesses that have been explored are related to the conditional independence assumption made by the HMM; that is, given the state sequence, observations generated by a HMM are temporally independent and identically distributed. Moderate improvements in the speech recognizer’s performance have been reported with use of such generalized speech models over the HMM, but this success has been typically limited to relatively small and simple speech recognition tasks (cf. Ostendorf et al., 1996; Deng & Aksmanovic, 1997; Rathinavelu & Deng, 1997).

Another important weakness of the HMM which has not been explored in detail until recently is its lack of ability to explicitly provide correlation across pronunciation units,
whereby the acoustics associated with one unit directly influences that of adjacent ones in a predictable manner. Such correlation is particularly strong for conversational or spontaneous speech that is characterized by a high degree of phonetic reduction. Since the previous versions of the generalized HMMs or stochastic segment models (e.g. those surveyed in Ostendorf et al., 1996) have not addressed this key weakness, it is not surprising to see that such models, in general, fail to provide significant benefit for conversational speech recognition.

In our earlier work reported in (Deng, 1998, 1999; Deng & Ma, 1999), we specifically addressed the problem of phonetic reduction associated with conversational speech in the design of a new type of statistical model of speech capable of capturing correlation across pronunciation units such as articulatory feature bundles and phones. In the work of Deng and Ma (1999) where the hidden vocal-tract-resonance (VTR) dynamics was used explicitly to embrace phonetic reduction, we found that the most important factor affecting the new recognizer’s performance is the boundaries of the dynamic regimes in the VTR model of speech. When the boundaries were manually adjusted from those automatically (often with gross errors) determined from a HMM system so as to conform to the dynamic regimes expected from the model, drastic reduction of recognition error in the Switchboard task was consistently observed. This motivates the current work aimed at developing 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 to be the likelihood on the observation data (Mel-frequency cepstral coefficient or 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 models such as HMM. This achieves the desirable goal of consistency modeling in the entire speech recognition system.

The remainder of this paper is organized as follows. In Section 2, we briefly review the statistical, VTR-based hidden dynamic model of speech which has been described in more detail in Deng (1998) and Deng and Ma (1999). In Section 3, the newly developed path-stack algorithm developed for automatic optimization of dynamic regimes in this statistical coarticulatory model for the VTR dynamics is presented in detail. Testing of the algorithm on real speech data in terms of the boundary estimation accuracy is reported in Section 4. Finally, Section 5 reports on improved performance in spontaneous speech recognition on the Switchboard corpus using an N-best rescoring paradigm. The performance improvement is measured in word error rate reduction by comparing the use of the path-stack algorithm and use of the dynamic regimes, heuristic set based on the phone segmentation computed from a state-of-the-art HMM system.

2. The VTR-based dynamic model of speech

2.1. Model formulation

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 earlier experiments are provided in Deng and Ma (1999) and Picone et al. (1999). This speech model, based on the partially hidden VTR dynamics, has been formulated in mathematical

1 Via many hours of spectrogram reading by an author. The process of doing this involved identification of initiation of formant transitions in the spectrogram associated with each phone in the speech utterance. Sometimes extrapolation was used where formants are masked by spectral zeros in the consonants or where the energy of the speech sounds is low.
terms as a constrained and simplified nonlinear dynamic system. This is a special version of
the general statistical hidden dynamic model described in Deng (1999) and Deng (1998).

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

\[
Z(k + 1) = \Phi Z(k) + (I - \Phi)T + W(k),
\]

where \( T \) and \( \Phi \) are the VTR target and “time constant”, respectively, both distinct for each
phone in a speech utterance (phone index omitted for clarity). \( W(k) \) is the additive sys-

tem noise modeled by an independent and identically distributed Gaussian process with zero
mean and covariance matrix \( Q \), intended to capture residual errors in the system dynamics of
Equation (1). \( I \) is the identity matrix.

Equation (1) applies to each of the phone-sized segments. In general, different segments
in a speech utterance have different parameters \( T \) and \( \Phi \), but the VTR state variables, \( Z(k) \),
are constrained to be continuous going from one segment to the next, accounting for cross-
segment dependence.

The observation equation in the dynamic system model is described by

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

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 independent and
identically distributed Gaussian process with zero mean and covariance matrix \( R \), intended to
capture residual errors in the mapping from \( Z(k) \) to \( O(k) \) of Equation (2). The multivariate
nonlinear mapping, \( h(Z) \), is implemented by a global multilayer perceptron (MLP).

This new speech model underlying our new speech recognizer reported in this paper is
based on new principles of modeling hidden dynamics of speech production in the VTR do-
main. This model consists of two separate components which accommodate separate sources
of speech variabilities. The first component, as described in Equation (1), is a smooth dy-
namic one, linear but nonstationary. The nonstationarity is described by left-to-right dy-
namic regimes corresponding to the phone sequence. The second component, as described
in Equation (2), is static and nonlinear. This component handles other types of variabili-
ties (lower level). The two components combined form a nonstationary, nonlinear dynamic
system whose structure and properties are well understood in terms of the general process
of human speech production and in terms of specific operations in a formant-based speech
synthesizer capable of producing speech utterances indistinguishable from natural speech
(Stevens, 1993, 1999).

2.2. Likelihood computation

Let us use \( O_1^N \) to represent a sequence of observations \( \{ O(1), O(2), \ldots, O(N) \} \) and \( \Theta \) repre-
sent a dynamic model. The log-likelihood of model \( \Theta \) producing observations \( O_1^N \) is defined
as

\[
L(O_1^N|\Theta) = \log p(O(1), O(2), \ldots, O(N)|\Theta)
= \sum_{k=1}^{N} \log p(O(k)|O_1^{k-1}, \Theta),
\]

where we adopt \( p(O(1)|O(0), \Theta) = p(O(1)|\Theta) \). For a linear dynamic system model,
\( p(O(k)|O_1^{k-1}, \Theta) \) would be a Gaussian distribution. For the nonlinear model as was used
in the current study, it is not a Gaussian distribution but it can be approximated by a Gaussian when the extended Kalman filter (EKF) is used for state estimation. When \( p(O(\theta)|O_1^{k-1}, \Theta) \) is approximated by a Gaussian, the log-likelihood can be shown to be

\[
L(O|\Theta) = -\frac{1}{2} \sum_{k=1}^{N} \left[ \log |\Sigma_{\tilde{O}_k}| + \tilde{O}_k \Sigma_{\tilde{O}_k}^{-1} \tilde{O}_k \right] + \text{const},
\]

where \( \tilde{O}_k \) is the pseudo-innovation sequence for the nonlinear model and \( \Sigma_{\tilde{O}_k} \) is its covariance. The pseudo-innovation sequence and its covariance are calculated during by the EKF.

For the VTR-based dynamic model of speech formulated in Equations (1) and (2), due to the use of nonlinearity in the measurement equation Equation (2), EKF is used for the pseudo-innovation computation. The general derivation and computational procedure of the EKF algorithm have been widely available in the literature; e.g. Kitagawa (1987); Bar-Shalom and Bar-Shalom (1988); Bar-Shalom and Li (1993); Mendel (1995) and Tanizaki (1996). In the work of speech recognizer design as reported in this paper, we have adapted the general EKF algorithm to take account of the target-directed constraint in the state dynamics as expressed in Equation (1). This adapted version of the EKF algorithm is summarized below, which has been implemented in this work:

\[
\hat{Z}_{k|k-1} = \Phi \hat{Z}_{k-1|k-1} + (I - \Phi)T
\]

\[
\Sigma_{k|k-1} = \Phi \Sigma_{k-1|k-1} \Phi^T + Q
\]

\[
\tilde{O}_k = O(k) - h(\hat{Z}_{k|k-1})
\]

\[
\Sigma_{\tilde{O}_k} = H_{k|k-1} \Sigma_{k|k-1} H_{k|k-1}^T + R
\]

\[
K_k = \Sigma_{k|k-1} H_{k|k-1}^T \Sigma_{\tilde{O}_k}^{-1}
\]

\[
\hat{Z}_{k|k} = \hat{Z}_{k|k-1} + K_k \tilde{O}_k
\]

\[
\Sigma_{k|k} = \Sigma_{k|k-1} - K_k \Sigma_{\tilde{O}_k} K_k^T.
\]

where \( \hat{Z}_{k|k-1} \) is the prediction of the state and \( \Sigma_{k|k-1} \) the prediction error covariance; \( \tilde{O}_k \) is the pseudo-innovation sequence and \( \Sigma_{\tilde{O}_k} \) its covariance; \( \hat{Z}_{k|k} \) is the filtered state value and \( \Sigma_{k|k} \) the filtering error covariance. \( K_k \) is the Kalman gain and \( H_{k|k-1} \) is the Jacobian matrix of function \( h(\cdot) \) computed at the point \( \hat{Z}_{k|k-1} \). As we are using an MLP for the nonlinear function \( h(\cdot) \), the Jacobian matrix can be computed by

\[
H_t(Z) = \frac{d}{dZ} h(Z) = [H_{ij}(Z)] = \begin{pmatrix}
\frac{\partial h_1}{\partial Z_1} & \frac{\partial h_2}{\partial Z_1} & \frac{\partial h_3}{\partial Z_1} \\
\frac{\partial h_1}{\partial Z_2} & \frac{\partial h_2}{\partial Z_2} & \frac{\partial h_3}{\partial Z_2} \\
\vdots & \vdots & \vdots \\
\frac{\partial h_1}{\partial Z_d} & \frac{\partial h_2}{\partial Z_d} & \frac{\partial h_3}{\partial Z_d}
\end{pmatrix},
\]

where each element in the matrix is

\[
H_{ij}(Z) = \sum_j W_{ij} \left[ \sum_l w_{jl} g(Z_l) \right] \left[ 1 - g \left( \sum_l w_{jl} g(Z_l) \right) \right] w_{jl}.
\]

In the above, \( w_{jl} \) denotes the MLP weights from input to hidden units and \( W_{ij} \) the MLP.

\( ^2 \) The EKF algorithm is also used for model parameter training, which has been described in detail in Deng and Ma (1999).
A path-stack algorithm for optimizing dynamic regimes

weights from hidden to output units, where \( l \) is the index to an input node, \( j \) to a hidden node, and \( i \) to an output node. \( g(\cdot) \) is the standard sigmoid function:

\[
g(x) = \frac{1}{1 + \exp(-x)}.
\]

3. The path-stack algorithm for optimizing dynamic regimes

In order to motivate the development of the path-stack algorithm as the principal contribution of this paper, in this section we explain first why the standard dynamic programming approach (Viterbi algorithm) cannot be applied directly to search for optimal dynamic regimes in the hidden dynamic model of speech described in Section 2, no matter how small the number of the dynamic regimes is. In short, the difficulty for the search efficiency arises because of the continuity constraint imposed across dynamic regimes in the hidden variable domain. A consequence of this continuity constraint is this: at any fixed node in the trellis search diagram, the local likelihood scores, computed according to Equation (4), into a fixed future node will, in general, be different depending on their past history prior to arriving at the current node.\(^3\) 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 to optimizing the dynamic regimes in computing the global likelihood for a speech utterance consisting typically of many dynamic regimes.

3.1. Motivation of the algorithm: An example

In the trellis depicted in Figure 1 we show three left-to-right dynamic regimes \( (S_1, S_2, \text{and } S_3) \) for simplicity reasons. At time frame \( t = 3 \), there are two possible paths entering regime \( S_2^{\prime} \): “b11 → b22 → b23”, and “b11 → b12 → b23”. Note that at time \( t = 2 \) these two paths have an identical value, \( \hat{Z}_{1,1}, \Sigma_{1,1} \), to initialize the EKF. However, since the two paths use distinct model parameters \( (T, \Phi) \), 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, \( \hat{Z}_{1,2,2}, \Sigma_{1,2,2} \) and \( \hat{Z}_{2,2,2}, \Sigma_{2,2,2} \). 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, \( \hat{Z}_{1,2,2}, \Sigma_{1,2,2} \) and \( \hat{Z}_{2,2,2}, \Sigma_{2,2,2} \), 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 \( \hat{Z}_{1,3,3}, \Sigma_{1,3,3} \) and \( \hat{Z}_{2,3,3}, \Sigma_{2,3,3} \), at frame \( t = 3 \) and for regime \( (S_2) \).

In Viterbi algorithm for the conventional HMM, at this point the path with the lower likelihood of 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.\(^4\)

However, if we were to drop the low-score path at node “b23” as in the Viterbi algorithm for the 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

\[^3\] It is the difficulty that arises from such a constraint type which prevents further development of a model that is also substantially different from the conventional HMM (Bakis, 1991). Note also that the computational complexity associated with most versions of stochastic segment models (Ostendorf et al., 1996) arises from very different causes where no explicit constraints across segments are imposed.

\[^4\] This is the essence of dynamic programming or Viterbi algorithm.
values, $\hat{Z}_{1,3|3}, \Sigma_{1,3|3}$ and $\hat{Z}_{2,3|3}, \Sigma_{2,3|3}$ for EKF) and would hence behave differently in the future path growth.

The problem exemplified above associated with the dynamic model and with the use of the EKF algorithm applies to essentially all nodes in the trellis. That is, due to the continuity constraint in the VTR domain from one dynamic regime to the next, no path drop shall be allowed 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.

Essential to this “path-stack” algorithm is our discovery that the initial filter error covariance, $\Sigma_{k|k}$, is of much less importance in determining likelihood scores than the initial filter value, $\hat{Z}_{k|k}$. It was also discovered that if the filtered values of two paths are close to each other, then the behavior of these two paths in the future will be very similar. In the example of Figure 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 from each other, then they will produce similar likelihood scores (using the same model parameters of $S_3$) when they enter node “b34” at time frame $t = 4$. Further, due to the asymptotic, target-directed property of the VTR dynamics modeled by Equation (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 two distinct paths occurring at an earlier time frame. Therefore, if we drop one (with a lower score) of these paths with similar filtered values at an early time frame, 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 frame, and limit the size of the path stack to a fixed value.\(^5\)

\(^5\)This is implemented by forcing the end value of the earlier regime to initialize the EKF for the newly entered regime.

\(^6\)Investigation of using an adaptive stack size is currently underway.
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 $m$th path ending at node $b$, $L(b, m, t = k)$, and calculate the filtered (by EKF) values, $\hat{Z}_{m,k|k}$ and $\Sigma_{m,k|k}$ (they are denoted by $Z(m, k)$ and $P(m, k)$, respectively, in Figure 2 for convenience). Then, compute the distance, $D_{mi}$, between $\hat{Z}_{m,k|k}$ 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 $\hat{Z}_{m,k|k}$ (say the $j$th path). If the path stack of node $b$ is not full and the distance $D_{mj}$ is greater than a preset threshold, insert the new path into the path stack. Otherwise, if the new path has a higher path score, substitute it for the earlier stored $j$th path, if not, drop the new path.

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 Figure 2. $K$ is the utterance length, $N$ is the total number of nodes in a lattice or in an $N$-best list, and $SS$ is the maximum number of paths that are allowed to be kept in a path stack.

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

The path-stack algorithm described above avoids the exponentially growing computation inherent in the exhaustive search by keeping only a limited but sufficient number of promising paths at each trellis node. It extends the earlier search algorithms and ideas developed mainly for the HMM-based recognizers, including the stack idea, the $N$-best search idea, and the idea of limiting the stack growth Bahl, Jelinek and Mercer (1983); Schwartz and Austin (1991); Gopalakrishnan, Bahl and Mercer (1995). Our specific contribution lies in applying these ideas to the specific speech model where different types of information are used to determine whether to grow, to maintain, or to delete the stack entries during the trellis search. The algorithm has been kept within the time-synchronous, dynamic programming framework.

4. Testing the algorithm on speech data

To examine the effectiveness of the algorithm and to confirm its correct implementation, we first tested the path-stack algorithm on simulated data. The results showed that the algorithm is indeed able to find the true boundaries for the simulated data. Further, building on the success of the algorithm on the simulation data, we carried out a series of experiments to test the path-stack algorithm on real speech data excised from the Switchboard corpora.

Since for the speech data we do not know what the true dynamic regime boundaries are, the best we can do to assess the accuracy of the algorithm is to compare the results of the algorithm with those from an exhaustive search.\(^7\)

We carried out an exhaustive search on the utterances by allowing the stack growth to be exponential with time frames (i.e. no path deletion and pruning in the path-stack algorithm for dynamic programming). The results of the exhaustive search for three typical utterances, marked as stack size equal to $\infty$, are listed in Table I. The results are expressed as the log-likelihood of the full path at the end of the search. The results for the same utterances obtained

\(^7\) We used a brute-force method for exhaustive search where the regime boundaries are hypothesized at each possible frame and the utterance likelihoods for all these hypothesized boundaries are computed and compared.
For each frame, $O(k)$, in the observation sequence of an utterance ($0 < k < K + 1$)
best_score ($k$) = negative infinity;
For each node (phone) $i$ in the lattice (or in an $N$-best list) of the utterance ($0 < i < N + 1$)
Let $n_{\text{path in stack of node, } i} = 0$;
For each node $j$ (not pruned) which can enter node $i$ ($0 < j < N + 1$)
For each path $m$ existing in the path stack of node $j$ at time $k - 1$ ($0 < m < SS + 1$)
Calculate acoustic score $L_j(O(k)|j, m)$, filtered dynamics $Z(m, k)$ and its
error covariance $P(m, k)$;
Let $\text{min_distance} = \infty$ and $\text{min_path_index} = 0$;
For each path $l$ existing in the path stack of node $i$ at time $k$ ($0 < l < SS + 1$)
Let distance $= |Z(i, l, k) - Z(m, k)|$;
if (distance < min_distance)
\text{min_distance} = distance;
\text{min_path_index} = $l$;
endif
End
Let $\text{path_likelihood} = L(j, m, k - 1) + L_j(O(k)|j, m)$;
if ( $n_{\text{path in stack of node, } j} > SS$ and $\text{min_distance}/|Z(m, k)| > \text{Thres_deletion}$ )
$L(i, n_{\text{path in stack of node, } j}, k) = \text{path_likelihood};$
$Z(i, n_{\text{path in stack of node, } j}, k) = Z(m, k);$  
$P(i, n_{\text{path in stack of node, } j}, k) = P(m, k);$  
$n_{\text{path in stack of node, } j}^{++} = n_{\text{path in stack of node, } j}^{++} + 1$;
Remember the back pointer from ($i, n_{\text{path in stack of node, } j}$) to ($j, m$);
else if ($L(i, \text{min} \text{path_index}, k) < \text{path_likelihood}$)
$L(i, \text{min} \text{path_index}, k) = \text{path_likelihood}$
$Z(i, \text{min} \text{path_index}, k) = Z(m, k);$  
$P(i, \text{min} \text{path_index}, k) = P(m, k);$  
Re-set the back pointer for node ($i, n_{\text{path in stack of node, } j}$) to ($j, m$)
else
Drop the new path and keep the stack untouched;
endif
if ( $\text{path_likelihood} > \text{best_score} (k)$ )
Let best_score ($k$) = path_likelihood;
endif
End
End
Do the following path pruning if needed.
For each node $i$ in the lattice and each path $k$ in the path stack of node $i$
if ( $|(\text{best_score} (k) - L(i, m, k)/\text{best_score} k)| > \text{Thres_pruning}$ )
Prune path ($i, m$) from the search space;
endif
End
End
Choose the path with maximum likelihood. maxpathlikelihood = max $L(i, m, k)$ over node $i$
and path $k$ in the stacks.
Trace back to obtain the segmentation of dynamics and its associated speech units.

Figure 2. The path-stack search algorithm.
A path-stack algorithm for optimizing dynamic regimes

Table I. Log-likelihoods of the exhaustive search and of the path-stack algorithm (stack size of 20)

<table>
<thead>
<tr>
<th>Stack size</th>
<th>2151A18 (frames-in-phone)</th>
<th>2121A19</th>
<th>2335B14</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 frames, 3 phones</td>
<td>2151A18 (frames-in-phone)</td>
<td>2121A19</td>
<td>2335B14</td>
</tr>
<tr>
<td>inf</td>
<td>-71004.6 (1 6.5)</td>
<td>-33577.0</td>
<td>-39348.8</td>
</tr>
<tr>
<td>20</td>
<td>-71004.6 (1 6.5)</td>
<td>-33577.0</td>
<td>-39348.8</td>
</tr>
</tbody>
</table>

by the path-stack algorithm with a stack size equal to 20 are also shown in the same table for comparison purposes. While applying the path-stack algorithm, the threshold for path deletion on path stacks, denoted as “Thres_deletion” in Figure 2, was set to 0.01, which was empirically determined from the experiments on the simulation data. No path pruning at each time frame was imposed; i.e. setting the pruning threshold, denoted as “Thres_pruning” in Figure 2, to a very large value. The identical log-likelihoods of the exhaustive search and of the path-stack algorithm indicate that the path-stack algorithm is indeed able to find the globally optimal segment boundaries or dynamic regimes.

The first utterance in Table I, named “2151A18”, contains only three phones (three regimes with one for each phone) in “2151A18” (“b”, “ah” and “t”) or three regimes (with one for each phone), and it needs only two segment boundary values to delineate the three dynamic regimes. Hence it is possible to list all possible boundary values—optimal as well as nonoptimal ones. All the possible boundary values which have been used in the exhaustive search are listed in Table II. The globally optimal boundaries (i.e. the ones which give the largest utterance log-likelihood of -71004.6) correspond to the phone durations of 1, 6, and 5, respectively. This is marked by “*” in Table II.

We now turn to a study of the effects of deletion threshold (Thres_deletion) and stack size on the CPU time and log-likelihood for the path-stack algorithm, where the results are shown in Table III for two typical utterances (named “U1” and “U2”, which represent Switchboard utterances, “2121A-0019” and “2335B-0014”, respectively). These parameters in the algorithm are chosen to provide a compromise between the search speed and the search optimality. In producing these results, we set “Thres_pruning” to a very large value so that no path pruning at any time frame occurs. This allows us to explore how a combination of path stack size and “Thres_deletion” influences the search performance. The CPU time was calculated by running the algorithm on a “Sun Ultra 10” computer. The results shown in Table III suggest that a combination of a path stack size of 5 and “Thres_deletion” of 0.02 gives satisfactory compromise, where the path-stack algorithm does not lose optimality while saving a significant amount of CPU time.

By fixing “Thres_deletion” to 0.02, similar results from three new utterances with widely varying lengths are obtained, which are shown in Table IV. Again, a path stack size of 5 appears to be sufficient to have produced the optimal search result (optimal regime boundaries), as evidenced by the converging likelihood scores.

It surprises us somewhat that a very small path stack size is often sufficient for a successful search for the best path which gives the globally optimal likelihood for the VTR-based dynamic model of speech presented in Section 2. It is interesting to analyse why this has been the case. It appears that the dynamic model’s target-directed asymptotical property accounts for much of the success of the path-stack algorithm while requiring only a small stack size.

---

8 It has been impractical to perform an exhaustive search for long utterances due to the large number of dynamic regimes.
9 The trained $\Phi$ values are usually between 0.2 and 0.9 in our experiments.
A path-stack algorithm for optimizing dynamic regimes

Table IV. Further results of the path-stack algorithm exploring the effect of the stack size on the utterance log-likelihood value

<table>
<thead>
<tr>
<th>Stack size</th>
<th>2151A18</th>
<th>2151A24</th>
<th>2229A70</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-71.110.4</td>
<td>-22.718.6</td>
<td>-96.335.9</td>
</tr>
<tr>
<td>2</td>
<td>-71.004.6</td>
<td>-22.718.6</td>
<td>-96.335.9</td>
</tr>
<tr>
<td>3</td>
<td>-71.004.6</td>
<td>-22.718.6</td>
<td>-96.454.1</td>
</tr>
<tr>
<td>5</td>
<td>-71.004.6</td>
<td>-22.718.6</td>
<td>-96.432.4</td>
</tr>
<tr>
<td>10</td>
<td>-71.004.6</td>
<td>-22.718.6</td>
<td>-96.432.4</td>
</tr>
<tr>
<td>50</td>
<td>-71.004.6</td>
<td>-22.718.6</td>
<td>-96.432.4</td>
</tr>
</tbody>
</table>

Table V. Effects of pruning threshold on CUP time and log-likelihood (in parentheses) for the path-stack algorithm. Fixing the stack size = 5 and “Thres_deletion” = 0.02

<table>
<thead>
<tr>
<th>uttr</th>
<th>Thres_pruning = 1.0</th>
<th>Thres_pruning = 0.5</th>
<th>Thres_pruning = 0.25</th>
</tr>
</thead>
<tbody>
<tr>
<td>2121A19</td>
<td>65.0 s (−33.577.0)</td>
<td>45.0 s (−33.577.0)</td>
<td>36.0 s (−33.692.3)</td>
</tr>
<tr>
<td>2335B14</td>
<td>507 s (−39.348.0)</td>
<td>464.0 s (−39.348.8)</td>
<td>445.0 s (−39.348.3)</td>
</tr>
</tbody>
</table>

results we presented earlier in this section have demonstrated that the path-deletion strategy in the path-stack algorithm is reasonably effective while maintaining its efficiency.

Our final set of testing experiments concern the effects of path pruning on the optimality of the path-stack algorithm. We fix the stack size and “Thres_deletion” to be 5 and 0.02, respectively, and then vary the path-pruning threshold (“Thres_pruning”). The results for two typical utterances “2121A19” and “2335B14” are listed in Table V. For the first utterance, the algorithm begins to lose optimality when “Thres_pruning” is decreased to 0.25 while relatively little computation time is saved. For the second utterance, the algorithm appears to maintain optimality over a wider range of the stack-pruning threshold. This suggests that a value of 0.5 for “Thres_pruning” is most desirable.

5. Speech recognition experiments

In all the experiments reported in this paper, we used an N-best list rescoring paradigm (no language model was used during rescoring), according to the likelihood scoring algorithm in Equation (4), to evaluate the new recognizer based on the VTR dynamic model on the Switchboard speech data. The N-best list of word transcription hypotheses and their initial phone-level segmentation (i.e. alignment) are obtained from a state-of-the-art triphone-based HMM system (Bridle & Deng, 1998). The reasons for using the limited N-best rescoring paradigm in the current experiments are mainly due to computational ones. In all the experiments using the path-stack algorithm, we set the stack size, “Thres_deletion”, and “Thres_pruning” to be 5, 0.02 and 0.5, respectively, based on the testing experiments reported in Section 4.

One male speaker’s data (speaker ID: 1028) was extracted from the “train-ws97-a” training set in the Switchboard corpora and used for the training of the dynamic models. These data consist of 966 training utterances. One single MLP was trained for the entire recognizer, common to all phones.10

10 In our earlier work (Deng & Ma, 1999), we classified all the phones into 10 classes and each class was described by one separate MLP. This original design was intended to accommodate the physical reality that some consonants with a different manner of articulation may have similar VTR targets but have drastically different acoustic properties. In the development of decoding algorithms for the dynamic model of speech, however, we found that the original
TABLE VI. Recognition results (WERs) on the Switchboard data (480 utterances) with and without optimizing the dynamic regimes

<table>
<thead>
<tr>
<th>Two VTR systems</th>
<th>Ref+5</th>
<th>5-Best</th>
<th>Ref+100</th>
<th>100-best</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed dynamic regimes (by HMM)</td>
<td>29.5%</td>
<td>53.2%</td>
<td>50.3%</td>
<td>60.9%</td>
</tr>
<tr>
<td>Optimized dynamic regimes</td>
<td>26.3%</td>
<td>53.0%</td>
<td>47.0%</td>
<td>60.1%</td>
</tr>
</tbody>
</table>

During the recognizer testing, the \( N \)-best list where used to compare the VTR-dynamic recognizer, whose dynamic regimes are optimized by using the path-stack algorithm, with the earlier version of the VTR-dynamic recognizer using fixed dynamic regimes from the HMM phone segmentation. In these comparison experiments, a total of 480 test utterances are used. For the new recognizer, given the optimized dynamic regimes for each phone of each hypothesis in the \( N \)-best lists, the computation of the acoustic likelihoods is performed efficiently by running the EKF algorithm for each hypothesis in the \( N \)-best lists.

Table VI lists the word error rates (WERs) obtained from the comparison experiments, with (Ref+5 and Ref+100) and without (5-best and 100-best) adding the reference transcriptions to the \( N \)-best \( (N = 5 \) and \( N = 100 \) lists, for the present and the earlier versions of the VTR-based speech recognition system. The earlier version (row two) uses fixed dynamic regimes produced by a separate HMM system. The new version (row three) uses the path-stack algorithm to optimize the dynamic regimes and hence the likelihoods. Word error rate reduction of over an absolute value of 3% has been achieved, for the Ref+5 and Ref+100 cases, by using an implementation of the path-stack algorithm described in Section 3 with a very small stack size. For the other two cases without references included, the path-stack algorithm has produced very limited WER reduction over the earlier version of the recognition system based on HMM segmentation. The discrepancy that the recognition results are in favor of the new recognizer (as developed at the current stage) only when it is exposed to the reference transcription may be accounted for as follows. The long-span context dependency via the cross-segment constraint in the speech model trends to propagate the local phone errors in the nonreference transcription. This could severely affect the scores of the correct, adjacent phones some distance away. However, with reference transcriptions included, the above negative effects are completely eliminated, giving the model a greater chance to score high on correct phone transcriptions.

6. 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 reduction. The key property of the model is the continuity constraint in the hidden VTR state domain across pronunciation units. This highly desirable property for coarticulatory modeling, however, presents a special challenge for the computation of the global likelihood function over the optimized dynamic regimes governing local dynamics associated with each phone-sized speech unit. The path-stack algorithm presented in this paper reduces the computation complexity from that which is exponential in the utterance length (as in exhaustive search) to that which is only linear in the utterance length.

We have empirically evaluated the accuracy of the path-deletion procedure implemented design made the search almost unfeasible because separate boundaries for the dynamics and for the MLPs need to be simultaneously searched. Use of a single global MLP for all phones makes the algorithm development feasible. 11See detailed arguments for such desirability in Deng (1998).
in the path-stack algorithm by comparing the segmentation results (i.e., the optimized dynamic regimes) from the path-stack algorithm, using the Switchboard speech data where the true dynamic regimes are unknown, with those of exhaustive search where computation complexity is exponential in the utterance length. In many of the short utterances for which the exhaustive search is feasible, the path-stack algorithm is indeed able to produce consistent regimes with those from exhaustive search, as long as the noise levels are appropriately adjusted. For long utterances for which the exhaustive search is not feasible, we have obtained the desirable result of converging likelihoods with an increasing size of the path stack used in the algorithm.

The speech recognition evaluation results as reported in Section 5 are, although positive, have not yet been as striking as we had hoped for based on our earlier observation for the very low WER using hand-segmented dynamic regimes via spectrogram reading. One main reason is that modeling accuracy has not been high enough. In our future work, while we are striving for improvement of modeling accuracy, we will refine the algorithm in several aspects including the use of dynamically adapted stack sizes. We will also be investigating other possibilities for joint dynamic regime optimization, model parameter optimization, and noise-level adaptation.

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 Professor Fred Jelinek and Professor Mari Ostendorf for the encouragement, discussions, and support of this work, and also thank two anonymous reviewers whose suggestions greatly improved the quality of the paper presentation.

References


Stevens, K. *Speech Synthesis with a Formant Synthesizer*, Course Notes, special summer course 1993.


(Received 1 July 1999 and accepted for publication 23 December 1999)