AN EFFICIENT INCREMENTAL LIKELIHOOD EVALUATION FOR POLYNOMIAL TRAJECTORY MODEL USING WITH APPLICATION TO MODEL TRAINING AND RECOGNITION

Chak-Fai Li and Man-hung Siu

Electrical and Electronic Engineering Department
Hong Kong University of Science and Technology, Hong Kong

ABSTRACT

Polynomial Segment Model (PSM), which was first proposed in [1] and subsequently studied by other researchers [9] [8], has opened up an alternative research direction for speech recognition. In PSM, speech frames within a segment are jointly modeled such that any change in the boundaries of a segment would require the re-computation of the likelihood of all the frames within the segment. While estimation of the best segment boundaries are possible, the computation consideration typically constrains the PSM model to limit the search to center around some pre-segmentation typically obtained by using another model such as an HMM, in effect limiting the possibility of using PSM itself. In this paper we introduce a new approach to evaluate the likelihood of a PSM segment by efficiently "accumulating" segment likelihood incrementally, i.e. one frame at a time. Based on this incremental likelihood evaluation, an efficient PSM search and training algorithm are also introduced. We show the effectiveness of the incremental likelihood evaluation by building a PSM-based TIMIT recognition system (both training and test) without the need of using another model for pre-segmentation.

1. INTRODUCTION

While HMM is the most common representation for speech acoustics, the segmental models [2][3][**?**][4] have been pursued as alternatives that can better model the speech dynamics and the time correlations between speech frames. The polynomial segment model (PSM), first proposed by [1] as the polynomial trajectory model, represents the speech dynamics of a variable duration speech segment as a polynomial function. Previous works [1][**?**] have reported its superior performance as compared to using HMMs. Like other segmental models, in PSM, the likelihood of a segment is the joint likelihood of all the frames within it. Thus any change to the boundaries of a segment requires the re-computation of the likelihoods of all its frames which is computationally intensive. Because the computation required to search for the optimal segment boundaries is high, more limited searching methods, such as searching within a window of an HMM-generated segmentation or the use of the N-best re-scoring [5] techniques, are commonly used. Similarly, PSMs are often trained with pre-segmented data, either hand-segmented (as in the TIMIT corpus) or segmented using an HMM-based recognizer. These constraints make it difficult to build a self-contained PSM-based system and thus limits the potential use of PSM.

This work is partially funded by RGC CERG: project number HKUST6049/00E.

In this paper, we propose an efficient, incremental algorithm for computing the likelihood of a PSM segment such that the likelihood of a length N segment can be computed recursively from the likelihood of the length $N-1$ segment. Based on this incremental evaluation algorithm, efficient PSM search and training algorithms are proposed such that they do not require pre-segmentation by an HMM making the PSM system self-contained. In Section 2, we summarize the PSM formulation as described in [1]. Section 3 describes the proposed efficient incremental algorithm for computing the PSM segment likelihoods. In Section 4, we describe our PSM recognition and training algorithms. Preliminary experimental results are reported in Section 5 and we conclude the paper in Section 6.

2. POLYNOMIAL SEGMENT MODEL

Denote an N_k-frame speech segment, $C_{\tau_k}^{\tau_k+N_k-1} = (c[\tau_k], \ldots, c[\tau_k + N_k - 1])$, which starts at τ_k and ends at $\tau_k + N_k - 1$. Each frame in $C_{\tau_k}^{\tau_k+N_k-1}$is represented by a D dimensional vector, an $(R-1)$-th order PSM is given by $C_{\tau_k}^{\tau_k+N_k-1} = Z_k B_k + E_k$ where $C_{\tau_k}^{\tau_k+N_k-1}$ is a $N_k \times D$ observation matrix, B_k is the trajectory parameter matrix of dimension $R \times D$, Z_k is a $N_k \times R$ design matrix for time normalization and E is the residue error. For a quadratic (R=3) PSM, Z_k is given by

$$Z_k = \begin{bmatrix} 1 & 0 & 0 \\ 1 & \frac{1}{N_k-1} & \left(\frac{1}{N_k-1}\right)^2 \\ 1 & \frac{2}{N_k-1} & \left(\frac{2}{N_k-1}\right)^2 \\ \vdots & \vdots & \vdots \\ 1 & 1 & 1 \end{bmatrix}.$$

Z_k normalizes the segment length uniformly to one.

2.1. PSM Parameter Estimation

The maximum likelihood estimate of the trajectory parameter matrix of $C_{\tau_k}^{\tau_k+N_k-1}$is given by

$$B_k = \left[Z_k' Z_k\right]^{-1} Z_k' C_{\tau_k}^{\tau_k+N_k-1}.$$

The corresponding residue error covariance, Σ_k, is given by

$$\Sigma_k = \frac{E_k' E_k}{N_k} = \frac{(C_{\tau_k}^{\tau_k+N_k-1} - Z_k B_k)'(C_{\tau_k}^{\tau_k+N_k-1} - Z_k B_k)}{N_k}.$$

The triplet, $\{B_k, \Sigma_k, N_k\}$, can be viewed as the sufficient statistics for the $C_{\tau_k}^{\tau_k+N_k-1}$. For a set of K segments from model m, denoted as $S = \{C_{\tau_1}^{\tau_1+N_1-1}, \ldots, C_{\tau_K}^{\tau_K+N_K-1}\}$, the maximum likelihood estimate of the PSM parameter for model m, $\hat{B}_m$, and residue covariance, $\hat{\Sigma}_m$, are given by

$$\hat{B}_m = \left[\sum_{k=1}^{K} Z_k' Z_k\right]^{-1} \left[\sum_{k=1}^{K} Z_k' Z_k B_k\right],$$

and

$$\hat{\Sigma}_m = \frac{\sum_{k=1}^{K} (C_{\tau_k}^{\tau_k+N_k-1} - Z_k \hat{B}_m)' (C_{\tau_k}^{\tau_k+N_k-1} - Z_k \hat{B}_m)}{\sum_{k=1}^{K} N_k}.$$

2.2. Log Likelihood Evaluation

The likelihood of a segment against a model can be evaluated using its sufficient statistics as described in [1]. On the other hand, because a PSM represents the time-varying mean trajectory, the log likelihood of $C_{\tau_k}^{\tau_k+N_k-1}$ against model m, $L_{N_k}(C_{\tau_k}^{\tau_k+N_k-1}|\hat{B}_m, \hat{\Sigma}_m)$, can also be evaluated one frame at a time and is given by

$$L_N(C_{\tau_k}^{\tau_k+N_k-1}|\hat{B}_m, \hat{\Sigma}_m) = -\frac{N}{2}[D\log(2\pi) + \log|\hat{\Sigma}_m|] \qquad (1)$$
$$-\frac{1}{2} tr[(C_{\tau_k}^{\tau_k+N_k-1} - Z_k\hat{B}_m)\hat{\Sigma}_m{}^{-1}(C_{\tau_k}^{\tau_k+N_k-1} - Z_k\hat{B}_m)'].$$

To simplify our notation, we will drop the dependence on $\hat{B}_m, \hat{\Sigma}_m$ and use N instead of N_k, L_N instead of $L_{N_k}(C_{\tau_k}^{\tau_k+N_k-1}|\hat{B}_m, \hat{\Sigma}_m)$. Furthermore, we will assume $\tau_k = 1$. For a quadratic PSM, by expressing $\hat{B}_m = [\beta_1\beta_2\beta_3]'$,, we can re-write the log likelihood as

$$L_N = -\frac{N}{2}[D\log(2\pi) + \log|\hat{\Sigma}_m|] \qquad (2)$$
$$-1/2\sum_{i=1}^{N-1}\{[C[i] - \beta_1 - \beta_2(\tfrac{i}{N_k-1}) - \beta_3(\tfrac{i}{N_k-1})^2]$$
$$\times\hat{\Sigma}_m{}^{-1}[C[i] - \beta_1 - \beta_2(\tfrac{i}{N_k-1}) - \beta_3(\tfrac{i}{N_k-1})^2]'\}$$

While this re-written equation is not as compact as the matrix equation in Eqn 1, it highlights the exact dependence of the L_N on N_k through the $(\frac{i}{N_k-1})$ terms and is useful for deriving the incremental likelihood evaluation algorithm.

3. INCREMENTAL COMPUTATION OF SEGMENT LIKELIHOOD

In Equation 2, we notice that the L_N is dependent on N_k primarily through the $(\frac{i}{N_k-1})$ terms.

When one frame is appended to the segment C_1^N, denoted as $C_1[N+1]$ which is $N+1$ frames long, the evaluation of L_{N+1} will require the re-evaluation of $C[1], \ldots, C[N]$ because of the changes in time normalization. That is, L_N is not re-used. Ideally, we would like to re-write

$$L_{N+1} = q(C[N+1], N+1)L_N + r(C[N+1], N+1) \quad (3)$$

where $q(C[N+1], N+1)$ and $r(C[N+1], N+1)$ are functions that only depend on $C[N+1]$, and $N+1$. In this form, the log likelihood can be computed recursively. While we are unable to re-write L_{N+1} in the form of Eqn 3, we can rewrite $L_N = \sum_l w_{l,N}$ such that each term $w_{l,N}$ can be recursively computed in a similar form as in Eqn 3. The detail derivation for the quadratic PSM is shown and we then discuss how this can be generalized for higher order PSM.

3.1. Incremental Computation for Quadratic PSM

In Equation 2, we express the log likelihood of a quadratic polynomial trajectory in terms of $b_{m,j}$ and $(\frac{i}{N_k-1})$. This can be further expanded by multiplying out the quadratic term into the cross products of the parameters and the observations. Although the expanded equation is more complicated, it allows us to separate the terms that are dependent on the observations from those that only depend on the model. The log likelihood of the segment, L_N, can now be written as,

$$\begin{aligned} L_N = & -\tfrac{N}{2}[D\log(2\pi) + \log|\hat{\Sigma}_m|] \\ & - 1/2\textstyle\sum_{i=1}^{N-1}[(C[i]-\beta_1)\hat{\Sigma}_m^{-1}(C[i]-\beta_1)' \\ & - 2(C[i]-\beta_1)\hat{\Sigma}_m^{-1}\beta_2'(\tfrac{i}{N-1}) - 2(C[i]-\beta_1)\hat{\Sigma}_m^{-1}\beta_3'(\tfrac{i}{N-1})^2 \\ & + \beta_2\hat{\Sigma}_m^{-1}\beta_2'(\tfrac{i}{N-1})^2 + 2\beta_3\hat{\Sigma}_m^{-1}\beta_2'(\tfrac{i}{N-1})^3 \\ & + \beta_3\hat{\Sigma}_m^{-1}\beta_3'(\tfrac{i}{N-1})^4] \\ = & \; K_N + A_N + B_N + D_N + E_N + F_N + G_N. \end{aligned}$$

It is easy to see that the terms in the above formulation can be recursively computed using the following equations:

$$\begin{aligned} K_N &= -\tfrac{N}{2}(D\log(2\pi) + \log|\hat{\Sigma}_m|) \\ A_N &= -\tfrac{1}{2}\textstyle\sum_{i=0}^{N-1}(C[i]-\beta_1)\hat{\Sigma}_m^{-1}(C[i]-\beta_1)' \\ &= A_{N-1} - \tfrac{1}{2}(C[N-1]-\beta_1)\hat{\Sigma}_m^{-1}(C[N-1]-\beta_1)', \\ B_N &= -\tfrac{1}{2}\textstyle\sum_{i=0}^{N-1}(-2)(C[i]-\beta_1)\hat{\Sigma}_m^{-1}\beta_2'(\tfrac{i}{N-1}) \\ &= B_{N-1}(\tfrac{N-2}{N-1}) + (C[N-1]-\beta_1)\hat{\Sigma}_m^{-1}\beta_2', \\ D_N &= -\tfrac{1}{2}\textstyle\sum_{i=0}^{N-1}(-2)(C[i]-\beta_1)\hat{\Sigma}_m^{-1}\beta_3'(\tfrac{i}{N-1})^2 \\ &= D_{N-1}(\tfrac{N-2}{N-1})^2 + (C[N-1]-\beta_1)\hat{\Sigma}_m^{-1}\beta_3', \\ E_N &= -\tfrac{1}{2}\beta_2\hat{\Sigma}_m^{-1}\beta_2'\textstyle\sum_{i=0}^{N-1}(\tfrac{i}{N-1})^2 \\ &= E_{N-1}(\tfrac{N-2}{N-1})^2 - \tfrac{1}{2}\beta_2\hat{\Sigma}_m^{-1}\beta_2', \\ F_N &= -\beta_2\hat{\Sigma}_m^{-1}\beta_3'\textstyle\sum_{i=0}^{N-1}(\tfrac{i}{N-1})^3 \\ &= F_{N-1}(\tfrac{N-2}{N-1})^3 - \beta_2\hat{\Sigma}_m^{-1}\beta_3', \\ G_N &= -\tfrac{1}{2}\beta_3\hat{\Sigma}_m^{-1}\beta_3'\textstyle\sum_{i=0}^{N-1}(\tfrac{i}{N-1})^4 \\ &= G_{N-1}(\tfrac{N-2}{N-1})^4 - \tfrac{1}{2}\beta_3\hat{\Sigma}_m^{-1}\beta_3'. \end{aligned}$$

In the above formulation, because all the terms $A_N, B_N \ldots, K_N$ can be recursively computed, it is obvious that L_N, being a sum of them, can also be recursively computed. Because terms A_N, B_N, D_N are dependent on the observations, they have to be accumulated during the recognition process. For a 2-nd order PSM, accumulating these three terms may at the worst case increase the memory usage three times. With techniques such as caching, real storage can be smaller. The terms K_N, E_N, F_N, G_N are independent of the observations and thus can be pre-computed. K_N, E_N, F_N, G_N can either be pre-computed for each model and N or, by separately pre-computing the cross products of $\beta_i\hat{\Sigma}_m^{-1}\beta_j'$ for all i and j as well as the $\sum_{i=0}^{N-1}(\frac{i}{N-1})^k$ for all k and N, the K_N, E_N, F_N, G_N terms can be computed directly during recognition without recursion. The latter approach is more memory efficient.

3.2. Extension to Higher Order Polynomial

Extending the above formulation to higher order is quite straightforward. For an k-th order PSM, there are $K+1$ (3 for quadratic) terms that depend on the observations and have to be accumulated during recognition. In addition, there are $2K-1$ (3 for quadratic) terms that are products of the PSM parameters $\beta_i\hat{\Sigma}_m^{-1}\beta_j'$ and the powers of the $(\frac{i}{N_k-1})$, plus one term for the determinant of the covariance. These terms can be pre-computed as discussed in the quadratic case above.

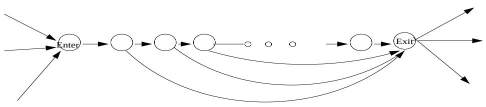

Fig. 1. A state-representation of the segment model

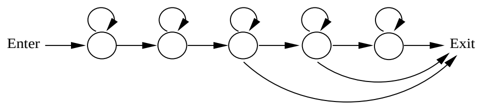

Fig. 2. A 5-state segment model with exit arcs

4. PSM RECOGNITION

While the incremental likelihood evaluation significantly reduce the computation in evaluating the segment likelihoods with the same starting points, to find the best segment path, we still need to consider all possible starting points for all phonetic units. In this section, we will first introduce one solution that makes use of the incremental likelihood computation to find the optimal path though it is computationally expensive. Then, we will formulate an approximation that is more computationally efficient.

Denote $O_1^T = o_1, \ldots, o_T$ as the observations of a sequence of phonetic segments $S_1^N = s_0^{t_1}, \ldots, s_{t_{N-1}}^{t_N}$ where $t_1, \ldots, t_N$ are the segment end times. The search will find $\hat{S}_1^N$ such that

$$(\hat{S}_1^N) = \arg\max_{S_1^N} P(O_1^T, S_1^N).$$

We use capital “I,J” to denote index for phonetic units.

Let’s consider a dynamic programming solution for finding $\hat{S}_1^N$. Define the quantity $\phi_i(t, m)$ to be the log likelihood of the best path up to time t and $s_m^t = I$, (i.e the last segment starts at time m and ends at time t is of model I). The path metric, $\phi_J(t)$, which is the log likelihood of the best path up to time t with the last segment is from model J can be computed as

$$\phi_J(t) = \max_m \phi_J(t, m). \quad (4)$$

If a segment begins at time t, the path metric depends on the previous phonetic units and is given by,

$$\phi_J(t, t) = \max_I [\phi_I(t-1) + \log a_{IJ}] + L(O_t^t|J),$$

where a_{IJ} is the transition probability of going from model I to J. For other segment begins ($m < t$), it can be extended from shorter segments.

$$\phi_J(t, m) = \phi_J(t-1, m) + \delta_J(O_t|s_m^{t-1}) \quad (5)$$

where

$$\delta_J(O_t|s_m^{t-1}) = L(O_m^t|J) - L(O_m^{t-1}|J)$$

is the likelihood change when extending the segment s_m^{t-1} (ends at $t-1$) to s_m^t (ends at t) which can be computed using the incremental likelihood evaluation described in Section 3,

The above equations can be represented by a state transition diagram as shown in Figure 1 similar to an HMM except that the transition costs are time-varying and dependent on the value stored in the states. An enter-state and an exit-state encompass a number of internal states. Let’s index the states by case “i,j”. The enter-state and exit states are similar to the null-nodes in an HMM. Each internal state represents a particular segment starting time $m_j(t)$. For a segment of phonetic unit J, the transition arcs into the enter-state have the costs of inter-unit costs $\log \alpha_{I,J}$. The transition costs between the states going to the t-th frame is given by:

$$\alpha_{i,j}(t) = \begin{cases} L(O_t^t|J) & \text{if } i = \text{enter-state} \\ 0 & \text{if } j = \text{exit-state} \\ \delta_J(O_t|s_{m_i(t-1)}^{t-1}) & \text{otherwise} \end{cases},$$

Because the connections between internal states and the exit state are null-arc t, there is no transition cost for going to the exit-state. Based on this state representation, we can apply an Viterbi-like algorithm using $\phi_j(t, m_j(t))$ as the the path metric. At each time instance, we need to update the starting time $m_j(t)$, the path metric $\phi_j(t, m_j(t))$ as well as introducing a trace-back pointer, $\psi_j(t)$.

While in Figure 1, each internal state has only one incoming arc, it is more general to assume that it can have more than one incoming arc which is the case for the exit-state. The updates of the internal states are given by

$$\begin{aligned} \psi_j(t) &= \arg\max_i [\phi_i(t-1, m_i(t-1)) + \alpha_{ij}(t)], \\ m_j(t) &= m_{\psi_j(t)}(t-1), \\ \phi_j(t, m_j(t)) &= \max_i \phi_i(t-1, m_i(t-1)) + \alpha_{ij}(t). \end{aligned}$$

The updates for the enter-state and exit-state are slightly different because the arcs going into these nodes are null-arcs that do not generate any observations but can be similarly derived. Furthermore, the initialization and termination are similar to that of a HMM.

If we use a very large number (upto the maximum number of frame per phone) of internal states, an optimal search is achieved. In the next section, we discuss how to modify the segment topology by using a smaller number of states.

4.1. Efficient Recognition Algorithm

In the above formulation, we need to keep a large number of states to represent all possible starting points for an optimal search. Instead, we can approximate the above state diagram by introducing self-loops in the states to reduce the number of states. In addition, we can include duration model probabilities in the transition arcs. In Figure 2, we showed a 5-state PSM that connects from state 3, 4 and 5 to the exit-state. This imposes a minimum segment duration of 3 frames while keeping 5 different segment starting times. While the structure of the segment model is now changed, the above algorithm and formulation are still valid. While the number of starting times are reduced, the maximal duration per phone is not restricted. Other topologies are also possible. By varying the number of states, we can trade-off modeling accuracy against speed and the experimental results will be discussed in the Section 5.

There is a small problem in the above approximation is that multiple states of the same segment may all have the same segment starting point. To avoid this and with minimal modification to the above algorithm, we can explicitly require the states of the same segment to keep paths with different segment starting points. As reported in [1], duration modeling has been shown to enhance segmental modeling performance. There are a number of ways to incorporate duration model information. Denote the probability of a length k segment of model J as $p_J(k)$. One approach is to apply the duration probability after a segment ended (and not extended) by adding it to the in-arcs of the exit state. In this approach, the transition cost of ending a segment after time t is given by:

$$\alpha_{i,\text{exit}}(t) = \log p_J((t) - m_i(t) + 1),$$

where $t - m_i(t) + 1$ is the duration of the ended segment at time t. The duration probability can also be computed during segment extension. In this approach, the transition cost, $\alpha_{ij}(t)$, of appending the t-th frame to a segment during a transition from state i to j is given by

$$\alpha_{ij}(t) = \delta_i(o_t|s^{t-1}_{m_i(t-1)}) + \log \frac{p_J(t - m_i(t-1) + 1)}{p_J((t-1) - m_i(t-1) + 1)}$$

The second approach has the advantage that the duration model information is being considered earlier during segment extension.

4.2. Training of PSM

The establishment of the PSM as a state representation in Figure 2 makes it possible to train PSM models by itself without pre-segmented data. The use of PSM training has an advantage that the segment alignments are consistent with that during test. While both the E-M algorithm and Viterbi training can be performed, the Viterbi training can greatly simplify the training implementation. For the results reported in the next section we used Viterbi training.

5. EXPERIMENTS

We performed two sets of experiments mainly to demonstrate the effectiveness of the incremental algorithm. All PSM experiments were performed using solely the PSM without any HMM model alignments or N-best re-scoring. The domain we focused on is phone recognition on the TIMIT corpus.

In our first set of experiment, we focused mainly on the relative speed of the recognition and the effect of varying the number of states. Our PSM model consisted of a single trajectory of second order (quadratic) with constant-time covariance and only 13 dimensional Mel-frequency cepstral coefficients (MFCC) with no derivative coefficients or advance techniques such as multiple mixtures or context-dependent modeling. We use the standard TIMIT training set and core test set [6] with SX and SI sentences, using the phone set in [7] plus three phones [el, en, and dx]. Bigram and duration modeling were used in this set of experiment. The weight for grammer, duration modeling and insertion penalty were tuned empirically. Our baseline system was a single mixture monophone HMM. The PSMs in this experiment were trained with the Viterbi style training stated above.

The experiment results are summarized in Table 1. The 1-state PSM's result is surprisingly good considering that only one segment begin-time was kept. The 3-state PSM in the 3rd row is a left-to-right model with self-loop on each state that enforces minimum duration of 3 frames which is the same state configuration as in the HMM. From the result, we can see that the addition of the number of states does improve performance but at a relatively slow rate. Imposing the constraint that states within the same segment which must have different frame begin gives a small gain. Increasing the number of states requires us to allow exit arcs to avoid creating an unreasonable minimum duration and the performance has further improved from the 3-state model. With 18 states, this is almost searching for all possible segment starting times but this performance is only slightly better than the 6-state PSM. The third column in Table 1 shows the relative computation speed relative to the HMM. Both the HMM and the PSM search code were not optimized and the numbers illustrate that the possibility of searching solely using PSM is within a reasonable amount of time. While the recognition accuracy is not very good because of the small feature set and lack of mixtures, this nevertheless demonstrated the feasibility of the incremental likelihood evaluation and the proposed search algorithm

Experiment	Phone Accuracy%	Speed
HMM	45.64	1
1-state	44.82	1.71
3-state	45.81	3.50
3-state w/ diff seg. starts	46.03	3.50
6-state w/ exit-arc	46.13	4.44
18-state w/ exit-arc	46.29	8.66

Table 1. Experimental result using different number of PSM states

In the second set of experiment, we use the same setting as before except that delta cepstral coefficients are introduced and number of state is fixed to 18. Using a single PSM recognition system, the accuracy is 49.2% compared with the 49.8% of 3-state HMMs. However, the number of parameters in the PSM models is only about 70% of that of the HMMs. Furthermore, because PSM is fitting a single trajectory across the phone with a single variance which is more constrained than an HMM, phonetic variations across speakers may be harder to capture with a single model. The addition of more mixtures should give a better performance. Training and testing with PSM mixtures is one direction we are currently pursuing.

6. CONCLUSION

In this paper, we have introduced an innovative approach to evaluate segment likelihood incrementally that makes it possible to perform model training and recognition solely using PSM. We also proposed an efficient search algorithm for PSM recognition and it can also be applied for segmental training. While the experiments reported are not at state-of-the-art performances, they illustrate the feasibility of building PSM-based system. The incremental algorithm, together with the search algorithm open up more possibilities for further development in PSM.

7. REFERENCES

[1] H. Gish and K. Ng, ”A Segmental Speech Model with Applications to Word Spotting,” Proc. ICASSP-93 pp.II-447-II-450, 1993

[2] M. Ostendorf, ”From HMM's to Segment Models: A Unified View of Stochastic Modeling for Speech Recognition,” IEEE Trans, SAP., vol 4, no. 5, pp. 360-378, 1996

[3] W. Holmes and M. Russell, ”Probabilistic-trajectory segmental HMMs,” Computer Speech and Language 13, pp. 3-37, 1999

[4] L. Deng, M. Aksmanovic, D. Sun, and J. Wu, ”Speech recognition using hidden Markov models with polynomial regression functions as nonstationary states,” IEEE Trans. SAP.,vol2,no.4, 1994

[5] M. Siu, R. Iyer, H. Gish and C. Quillen, “Parametric Trajectory Mixtures for LVCSR”, *Proc ICSLP*, 1998.

[6] V. Zue, S. Seneff, and J. Class. Speech Database Development at MIT: TIMIT and Beyond. Speech Communication, August 1990

[7] Kai-Fu Lee, ”Automatic Speech Recognition: The Development of the SPHINX System,” Kluwer Academic Publisher, 1989

[8] Y-S Yun and Y-H Oh, “A Segmental-feature HMM using parametric trajectory model”, *Proc ICASSP*, June, 2000.

[9] J. Lei and X. Bo, “Introduce segmental inner time warping into parameteri trajectory segment model for LVCSR”, *Proc ICSLP*, Sep 2002.