ESTIMATING TRAJECTORY HMM PARAMETERS USING MONTE CARLO EM WITH GIBBS SAMPLER

Heiga Zen, Yoshihiko Nankaku, Keiichi Tokuda, Tadashi Kitamura

Nagoya Institute of Technology
Department of Computer Science and Engineering
Gokiso-cho, Showa-ku, Nagoya 466-8555 Japan
{zen,nankaku,tokuda,kitamura}@ics.nitech.ac.jp

ABSTRACT

In the present paper, the Monte Carlo EM (MCEM) algorithm with a Gibbs sampler is applied for estimating parameters of a trajectory HMM, which has been derived from an HMM by imposing explicit relationships between static and dynamic features. The trajectory HMM can alleviate two limitations of the HMM, which are i) constant statistics within a state, and ii) conditional independence of state output probabilities, without increasing the number of model parameters. In a speaker-dependent continuous speech recognition experiment, trajectory HMMs estimated by the MCEM algorithm achieved significant improvements over the corresponding HMMs trained by the EM (Baum-Welch) algorithm.

1. INTRODUCTION

Speech recognition technology has achieved significant progress with the introduction of hidden Markov models (HMMs). Its tractability and efficient implementations are achieved by a number of assumptions, such as constant statistics within an HMM state, conditional independence assumption of state output probabilities. Although these assumptions make the HMM practically useful, they are not realistic for modeling sequences of speech spectra, especially in spontaneous speech. To overcome shortcomings of the HMM, a variety of alternative models have been proposed, e.g., [1–3]. Although these models can improve speech recognition performance, they generally require an increase in the number of model parameters and computational complexity. Alternatively, the use of dynamic features (delta and delta-delta features) [4] also improves the performance of HMM-based speech recognizers. It can be viewed as a simple mechanism to capture time dependencies. However, it has been thought of as an ad hoc rather than an essential solution. Generally, the dynamic features are calculated as regression coefficients from their neighboring static features. Therefore, relationships between static and dynamic feature vector sequences are deterministic. However, these relationships are ignored and the static and dynamic features are modeled as independent statistical variables in the HMM. Ignoring these dependencies allows inconsistency between the static and dynamic features when the HMM is used as a generative model in the obvious way.

Recently, a trajectory model, named a trajectory HMM, has been derived by reformulating the HMM whose state output vector includes both static and dynamic feature parameters [5]. The trajectory HMM can overcome the above limitations in the HMM framework without any additional parameters. In addition, the trajectory HMM provides a computational model for coarticulation and dynamics of human speech. A Viterbi-type (single path) training algorithm for the trajectory HMM has also been derived [5, 6]. However, the lack of an EM-type (multiple path) training algorithm does not permit marginalizing model parameters over hidden variables (state alignment). It may degrade model robustness and recognition performance.

In the present paper, the Monte Carlo EM (MCEM) algorithm [7] is applied for estimating trajectory HMM parameters. By using the MCEM algorithm, multiple paths sampled by the Markov Chain Monte Carlo (MCMC) method can be used to marginalize model parameters over hidden variables (state alignment).

The rest of the present paper is organized as follows. Section 2 defines the trajectory HMM. In Section 3, the Monte Carlo EM algorithm for the trajectory HMM is described. Results of a speaker-dependent continuous speech recognition experiment are shown in Section 4. Concluding remarks and future plans are presented in the final section.

2. TRAJECTORY HMM

The output probability of a sequence of static feature vector sequence $\boldsymbol{c} = \left[\boldsymbol{c}_1^\top, \ldots, \boldsymbol{c}_T^\top\right]^\top$ for a trajectory HMM Λ whose state output distributions are represented by mixtures of Gaussian components is given by

$$p(\boldsymbol{c} \mid \Lambda) = \sum_{\text{all } \boldsymbol{q}} p(\boldsymbol{c} \mid \boldsymbol{q}, \Lambda)\ P(\boldsymbol{q} \mid \Lambda), \tag{1}$$

$$p(\boldsymbol{c} \mid \boldsymbol{q}, \Lambda) = \mathcal{N}\left(\boldsymbol{c} \mid \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}\right), \tag{2}$$

where $\boldsymbol{c}_t$ is an M-dimensional acoustic static feature vector (e.g., MFCC, PLP, etc.) at time t, $\boldsymbol{q} = \{q_1, q_2, \ldots, q_T\}$ is a Gaussian component sequence, q_t is a Gaussian component at time t, and T is the number of frames in $\boldsymbol{c}$. In Eq. (2), $\bar{\boldsymbol{c}}_{\boldsymbol{q}}$ and $\boldsymbol{P}_{\boldsymbol{q}}$ are the $MT \times 1$ mean vector and the $MT \times MT$ covariance matrix corresponding to $\boldsymbol{q}$ given by

$$\boldsymbol{R}_{\boldsymbol{q}}\bar{\boldsymbol{c}}_{\boldsymbol{q}} = \boldsymbol{r}_{\boldsymbol{q}}, \tag{3}$$

$$\boldsymbol{R}_{\boldsymbol{q}} = \boldsymbol{W}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}\boldsymbol{W} = \boldsymbol{P}_{\boldsymbol{q}}^{-1}, \tag{4}$$

$$\boldsymbol{r}_{\boldsymbol{q}} = \boldsymbol{W}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}\boldsymbol{\mu}_{\boldsymbol{q}}, \tag{5}$$

$$\boldsymbol{\mu}_{\boldsymbol{q}} = \left[\boldsymbol{\mu}_{q_1}^{\top}, \ldots, \boldsymbol{\mu}_{q_T}^{\top}\right]^{\top}, \tag{6}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{q}} = \mathrm{diag}\left[\boldsymbol{\Sigma}_{q_1}, \ldots, \boldsymbol{\Sigma}_{q_T}\right], \tag{7}$$

where $\boldsymbol{\mu}_{q_t}$ and $\boldsymbol{\Sigma}_{q_t}$ are the $3M \times 1$ mean vector and the $3M \times 3M$ covariance matrix of the q_t-th Gaussian component, respectively, and $\boldsymbol{W}$ is a $3MT \times MT$ window matrix which appends first and second order time derivatives to the static feature vector sequence $\boldsymbol{c}$ as follows:

$$\Delta^{(1)}\boldsymbol{c}_t = \sum_{\tau=-L_-^{(1)}}^{L_+^{(1)}} w^{(1)}(\tau)\boldsymbol{c}_{t+\tau}, \quad \Delta^{(2)}\boldsymbol{c}_t = \sum_{\tau=-L_-^{(2)}}^{L_+^{(2)}} w^{(2)}(\tau)\boldsymbol{c}_{t+\tau}, \tag{8}$$

$$\boldsymbol{W} = [\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots, \boldsymbol{W}_T]^{\top} \otimes \boldsymbol{I}_{M\times M}, \tag{9}$$

$$\boldsymbol{W}_t = \left[\boldsymbol{w}_t^{(0)}, \boldsymbol{w}_t^{(1)}, \boldsymbol{w}_t^{(2)}\right], \tag{10}$$

$$\boldsymbol{w}_t^{(d)} = \Big[\underbrace{0, \ldots, 0}_{t-L_-^{(d)}-1}, w^{(d)}(-L_-^{(d)}), \ldots, w^{(d)}(0), \ldots, w^{(d)}(L_+^{(d)}), \underbrace{0, \ldots, 0}_{T-\left(t+L_+^{(d)}\right)}\Big]^{\top}, d = 0, 1, 2 \tag{11}$$

$L_-^{(0)} = L_+^{(0)} = 0$, and $w^{(0)}(0) = 1$.

Figure 1 shows an example of mean vector $\bar{\boldsymbol{c}}_{\boldsymbol{q}}$ and covariance matrix $\boldsymbol{P}_{\boldsymbol{q}}$ of a trajectory HMM. Model training conditions are the same as that of the Viterbi-trained trajectory HMMs described in Section 4. To obtain a Gaussian component sequence $\boldsymbol{q}$, a concatenated model composed of phoneme models /sil/, /a/, /i/, /d/, /a/, /sil/ was aligned to a natural speech using the delayed decision Viterbi algorithm [6]. Note that only elements corresponding to the first coefficient of mel-cepstrum are shown in the figure. It can be seen that not only the mean vector $\bar{\boldsymbol{c}}_{\boldsymbol{q}}$ varied in each state but also the inter-frame correlation was captured by the covariance matrix $\boldsymbol{P}_{\boldsymbol{q}}$. It is also interesting to note that the mean vector and inter-frame covariance corresponding to each monophone model changed according to its durations and neighboring models (see phoneme /a/). This shows that the trajectory HMM has a capability to capture coarticulation effects naturally.

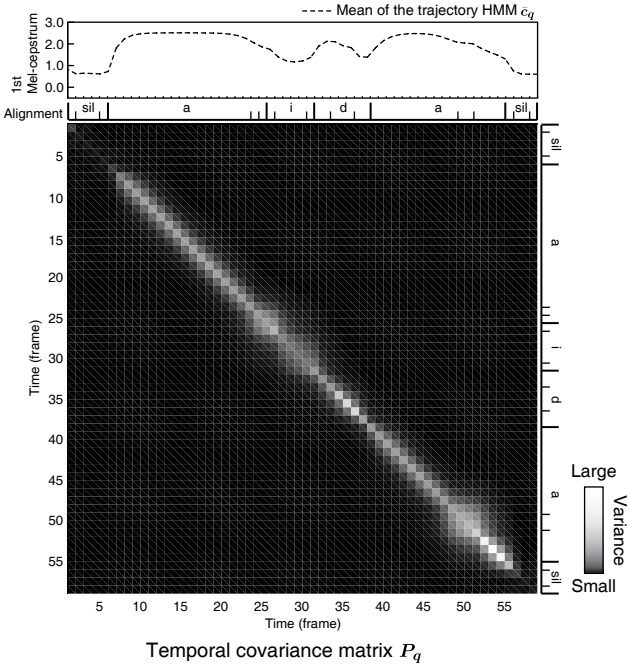

Fig. 1. An example of a trajectory HMM for a word "aida."

3. TRAINING ALGORITHM

The maximum likelihood criterion is used to estimate trajectory HMM parameters. In common with the HMM training, the EM algorithm may be used for ML training. An auxiliary function of a current parameter set Λ and a new parameter set Λ' is defined by

$$Q(\Lambda, \Lambda') = \sum_{\text{all } \boldsymbol{q}} P(\boldsymbol{q} \mid \boldsymbol{c}, \Lambda) \log p(\boldsymbol{c}, \boldsymbol{q} \mid \Lambda'). \tag{12}$$

Although it can be shown that by substituting Λ' which maximizes $Q(\Lambda, \Lambda')$ for Λ, model likelihood increases unless Λ is a critical point, exact EM training is prohibitive because the inter-frame covariance matrix is generally full and exact computation of $p(\boldsymbol{c} \mid \Lambda)$ or $P(\boldsymbol{q} \mid \boldsymbol{c}, \Lambda)$ has to be carried out over all possible Gaussian component sequences.

3.1. Viterbi approximation

To avoid the above problem, the Viterbi (single path) approximation has been applied [5]. By using this approximation, the auxiliary function can be expressed as follows:

$$\boldsymbol{q}' = \arg\max_{\boldsymbol{q}} P(\boldsymbol{q}' \mid \boldsymbol{c}, \Lambda), \tag{13}$$

$$Q(\Lambda, \Lambda') \approx \log p(\boldsymbol{c}, \boldsymbol{q}' \mid \Lambda'). \tag{14}$$

By taking the partial derivative of Eq. (14) with respect to model parameters, reestimation formulas for the Viterbi

training has been derived [5]. Although searching the optimal Gaussian component sequence $\boldsymbol{q}'$ is still intractable, the delayed decision Viterbi algorithm to find a better sub-optimal Gaussian component sequence has also been proposed [6].

3.2. Monte Carlo EM algorithm

It is generally considered that the single path approximation would be too strict for modeling spontaneous speech. Furthermore, we are unable to marginalize model parameters over hidden variables (Gaussian component sequence). This may degrade model robustness and recognition performance. In the present paper, the above approximation is relaxed using multiple Gaussian component sequences sampled by the MCMC method. This case is known to as the MCEM algorithm [7]. Equation (12) can be approximated by N Gaussian component sequences as follows:

$$\boldsymbol{q}^{(n)} \sim P(\boldsymbol{q} \mid \boldsymbol{c}, \Lambda) \quad 1 \leq n \leq N, \tag{15}$$

$$Q(\Lambda, \Lambda') \approx \sum_{n=1}^{N} \frac{1}{N} \log p\left(\boldsymbol{c}, \boldsymbol{q}^{(n)} \mid \Lambda'\right), \tag{16}$$

where $\boldsymbol{q}^{(n)}$ is the n-th Gaussian component sequence sampled by the MCMC method. There are several instances of the MCMC method, a Gibbs sampler is employed in the present paper. The Gibbs sampling algorithm for the trajectory HMM can be summarized as follows:

1. initialize $\boldsymbol{q}^{(1)} = \left\{q_1^{(1)}, \ldots, q_T^{(1)}\right\}$;
2. for iteration $2 \leq n \leq N$,
   draw samples $q_t^{(n)} \sim P\left(q_t \mid \boldsymbol{c}, \boldsymbol{q}_{-t}^{(n)}, \Lambda\right)$, where
   $\boldsymbol{q}_{-t}^{(n)} = \left\{q_1^{(n)}, \ldots, q_{t-1}^{(n)}, q_{t+1}^{(n-1)}, \ldots, q_T^{(n-1)}\right\}$.

By taking the partial derivative of the approximated auxiliary function with respect to model parameters in the same manner used in the Viterbi training, reestimation formulas for the MCEM training can be derived.

The performance of this method was evaluated in a preliminary experiment. Unfortunately, its results were much worse than that of the Viterbi training. In this experiment, rescoring paradigm using the delayed decision Viterbi algorithm was used. In this algorithm, the Gaussian component q_t maximizing $p(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{t+J}, q_1, \ldots, q_{t+J} \mid \Lambda)$ is selected in a time-recursive manner. Effects of determination of q_t to its neighboring frames can be incorporated. As a result, a better sub-optimal Gaussian component sequence can be obtained. On the other hand, the Gibbs sampler can sample q_t using full dependency between $\boldsymbol{c}$ and $\boldsymbol{q}$. Although this could be the advantage of the sampling method, we expect that this inconsistency between the training and decoding might degrade the recognition performance.

To relax this inconsistency, dependency between $\boldsymbol{c}$ and $\boldsymbol{q}$ are limited during sampling a Gaussian component. Instead of full posterior probability distribution, q_t is sampled from posterior probability distribution depending only on past, current and future J observations. The sampling algorithm is summarized as follows:

1. initialize $\boldsymbol{q}^{(1)} = \left\{q_1^{(1)}, \ldots, q_T^{(1)}\right\}$;
2. for iteration $2 \leq n \leq N$
   draw samples $q_t^{(n)} \sim P\left(q_t \mid \boldsymbol{c}_1, \ldots, \boldsymbol{c}_{t+J}, \boldsymbol{q}_{-t}^{(n)}, \Lambda\right)$, where
   $\boldsymbol{q}_{-t}^{(n)} = \left\{q_1^{(n)}, \ldots, q_{t-1}^{(n)}, q_{t+1}^{(n-1)}, \ldots, q_{t+J}^{(n-1)}\right\}$.

4. EXPERIMENT

4.1. Experimental conditions

Phonetically balanced 503 sentences uttered by a speaker MHT from the ATR Japanese speech database b-set was used. The first 450 sentences were used for training context-independent HMMs and trajectory HMMs. The remaining 53 sentences were used for evaluation. These test utterances had an average length of 43 phonemes and an average duration of 4 seconds.

Speech signals were sampled at a rate of 16 kHz and windowed by a 25.6-ms Blackman window with a 10-ms shift, and then mel-cepstral coefficients were obtained by a mel-cepstral analysis technique. Static feature vectors consisted of 19 mel-cepstral coefficients including the zeroth coefficient. They were augmented by appending their first and second order dynamic features. These dynamic features were calculated using Eq. (8) with $L_-^{(1)} = L_+^{(1)} = L_-^{(2)} = L_+^{(2)} = 1$, $w^{(1)}(-1) = -0.5$, $w^{(1)}(0) = 0.0$, $w^{(1)}(1) = 0.5$, $w^{(2)}(-1) = 1.0$, $w^{(2)}(0) = -2.0$, $w^{(2)}(1) = 1.0$.

Three-state left-to-right structure was used for modeling 36 Japanese phonemes including silence and short pause. Each state has a Gaussian component with a diagonal covariance matrix.

First, HMMs were initialized by the segmental k-means algorithm and reestimated using the EM (Baum-Welch) algorithm. Then, trajectory HMMs were iteratively trained by the Viterbi training (Viterbi), the MCEM training with 10 samples (MCEM10) and 50 samples (MCEM50), using the HMMs as their initial models. We constructed $3 \times 4 \times 5$ models by changing the number of delay J from 2 to 5 in the delayed decision Viterbi algorithm and Gibbs sampling. In the MCEM training, the initial Gaussian component sequences $\boldsymbol{q}^{(1)}$ were given by the delayed decision Viterbi algorithm. The same number of delay J was used both in the delayed decision Viterbi algorithm and the Gibbs sampling.

4.2. Experimental results

In all recognition experiments reported in this section, 100-best rescoring paradigm was used. A 100-best list was generated for each test utterance using the HTK Viterbi decoder with the HMMs used as the initial models for training the trajectory HMMs. Then, each hypothesis was resegmented and rescored by the delayed decision Viterbi algorithm with

Table 1. Phoneme error rates (%) for the 100-best lists re-scoring using the trajectory HMMs (reference transcriptions were *not included* in the re-scored hypotheses).

Training	Delay	#Iteration				
		1	2	3	4	5
Viterbi	2	18.4	18.4	18.3	18.4	18.4
	3	18.5	18.3	**18.0**	18.5	18.5
	4	18.1	18.3	18.4	18.1	18.1
	5	18.3	18.2	**18.0**	18.1	18.1
MCEM10	2	18.7	18.6	19.0	18.1	18.8
	3	18.5	17.9	18.3	18.5	18.5
	4	18.2	**17.8**	18.4	**17.8**	18.3
	5	18.4	18.4	18.4	18.3	18.3
MCEM50	2	18.7	18.3	18.9	18.8	18.7
	3	18.3	18.4	18.4	18.4	18.4
	4	18.1	**17.6**	18.5	18.3	18.2
	5	17.9	18.2	18.3	18.1	18.4

Table 2. Phoneme error rates (%) for the 100-best re-scoring using the trajectory HMMs (reference transcriptions were *included* in the re-scored hypotheses).

Training	Delay	#Iteration				
		1	2	3	4	5
Viterbi	2	10.6	11.4	9.4	**9.0**	9.8
	3	11.7	10.8	11.0	10.3	10.6
	4	10.8	9.1	9.1	10.3	9.5
	5	11.5	10.2	10.2	9.7	9.6
MCEM10	2	11.9	10.4	11.9	10.2	10.7
	3	11.1	10.1	9.1	10.3	10.2
	4	11.5	10.9	**8.9**	9.5	**8.9**
	5	10.9	9.7	9.7	10.2	9.6
MCEM50	2	12.1	12.2	11.6	10.9	10.3
	3	10.6	10.5	10.3	10.0	10.4
	4	11.4	10.3	10.4	9.9	**9.0**
	5	11.8	12.4	11.0	11.3	11.6

beam search (beam width = 1500). In this experiment, the same number of delay J was used both in training and decoding. To give an idea of the range of these 100-best lists, the error rates of the baseline HMMs, best, worst, and average of randomly selected hypotheses (100 times) were 19.7%, 13.9%, 27.4%, and 21.2%, respectively.

Tables 1 and 2 show the phoneme error rates for rescoring the 100-best hypotheses with and without reference transcriptions, respectively. All of the trajectory HMMs outperformed the baseline HMMs. When reference transcriptions were not included, the trajectory HMMs with 4 reestimates and delay of $J = 4$ by the MCEM algorithm using 50 samples achieved the best result. Compared with the best results of the Viterbi-trained trajectory HMMs and the baseline HMMs, about 3% and 11% error reductions were achieved, respectively. When reference hypotheses were included, the trajectory HMMs with 3 or 5 reestimates and delay of $J = 4$ by the MCEM algorithm using 10 samples achieved the best result. When reference transcriptions were included, about 55% error reduction over the HMM was achieved. However, there was not statistically significant difference between the Viterbi-trained and MCEM trained trajectory HMMs.

Compared with the Viterbi-trained trajectory HMMs, the performance of the MCEM-trained ones were unstable. In Tab.1, phoneme error rates of the Viterbi-trained trajectory HMMs ranged from 18.0% to 18.5%. On the other hand, that of the MCEM-trained ones ranged from 17.6% to 19.0%. It may be caused by that the MCEM is based on a stochastic method. Inconsistency between decoding and training (sampling method was used only on training) could be another reason of this phenomenon.

5. CONCLUSION

In the present paper, the Monte Carlo EM algorithm with a Gibbs sampler was applied for estimating parameters of a trajectory HMM. In a speaker-dependent continuous speech recognition experiment using rescoring scheme, trajectory HMMs trained by the MCEM algorithm achieved about 11% relative error reduction over the corresponding HMMs.

Future work includes implementation of a Viterbi decoder and evaluation in more practical conditions.

6. REFERENCES

[1] M. Ostendorf, V. Digalakis, and O.A. Kimball, "From HMMs to segment models," *IEEE Transactions on Speech & Audio Processing*, vol. 4, no. 5, pp. 360–378, 1996.

[2] A.V.I. Rosti and M.J.F Gales, "Switching linear dynamical systems for speech recognition," Tech. Rep. CUED/F-INFENG/TR.461, Cambridge University, 2003.

[3] G. Zweig, *Speech recognition using dynamic Bayesian networks*, Ph.D. thesis, University of California, Berkeley, 1998.

[4] S. Furui, "Speaker independent isolated word recognition using dynamic features of speech spectrum," *IEEE Transactions Acoustics, Speech, & Signal Processing*, vol. 34, pp. 52–59, 1986.

[5] K. Tokuda, H. Zen, and T. Kitamura, "Trajectory modeling based on HMMs with the explicit relationship between static and dynamic features," in *Proc. of Eurospeech*, 2003, pp. 865–868.

[6] H. Zen, K. Tokuda, and T. Kitamura, "A Viterbi algorithm for a trajectory model derived from HMM with explicit relationship between static and dynamic features," in *Proc. of ICASSP*, 2004, pp. 837–840.

[7] G.C.G. Wei and M.A. Tanner, "A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms," *Journal of the American Statistical Association*, vol. 85, pp. 699–704, 1990.