TR-IT-0132

近似反復 Bayes 推論に基づく連続密度 Hidden Markov Model の逐次適応学習

村上 雄大 Katsuhiro Murakami
ホウ シャン Qiang Huo
シンガー ハラルド Harald Singer

1995.09

この報告では、一回の発話ごとにその一回の発話データのみを次の適応データとして用いる逐次話者適応として、quasi-Bayes 学習法を取り上げ、そのインプリメントについて考える。また従来の適応法の比較実験について紹介する。

目次

1 イントロダクション

音声認識を行なう場合、同一話者でも周囲の環境が変わると認識率が悪くなる。また話者が変わると、同様に認識率が低下するため、何らかの話者適応が必要である。従来の MAP を用いた話者適応では、認識の前に、あるまとまった量の適応サンプルが必要であり、それらを入力することはユーザの側からすると煩わしい。また新たな適応データを追加する場合、全データを用いた再度の学習が必要であるため、マシンへの負担が大きい等の問題がある。

この報告では、これらの問題を解決するために、一回の発話ごとにその一回の発話データのみを次の適応データとして用いる逐次話者適応として、 quasi-Bayes 学習法 [2] を取り上げ、そのインプリメントについて考える。また従来の適応法の比較実験について紹介する。

具体的には、ガウス混合連続密度 hidden Markov model(CDHMM) の逐次適応学習の問題を取り上げる。

シナリオはおおよそ次の通りである。学習済みの CDHMM ベースの音声認識システムで始める。新しい話者が特定のタスクに対してシステムを使うのに、それぞれの発話後に認識システムが、ユーザが話す現在の発話のみを用いて適応的に学習される。アップデイトされたモデルが、次の発話を認識するのに使われる (図 1)。

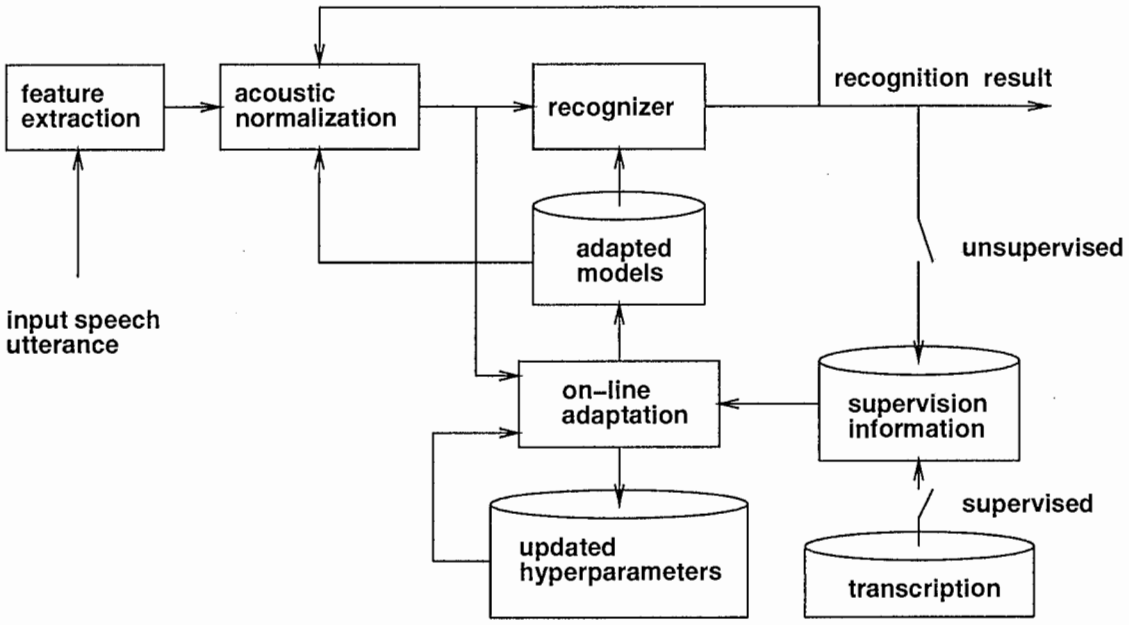

図 1: Block diagram of on-line Bayesian adaptation of HMMs

2 Bayes 学習法

2.1 反復 Bayes 学習法

今、$\chi^n = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_n\}$ を、CDHMM パラメータ λ を推定するのに使われる n 個の独立な出現サンプル (発話) とする。λ の正式な Bayes 推論は次の式に基づいている。

$$p(\lambda \mid \chi^n) = \frac{p(\chi^n \mid \lambda) \cdot p(\lambda)}{\int_\Omega p(\chi^n \mid \lambda) \cdot p(\lambda) d\lambda} \tag{1}$$

ここで Ω はパラメータ空間の許容領域である。λ の最大事後確率推定 MAP(maximum aposteriori) の解は、EM(estimate-maximize) アルゴリズムを用いることにより得られる。

サンプル $\mathbf{X}_i$ が一つずつ連続的に与えられるとすると、Bayes の定理を用いて、χ^n を与えたときの事後確率密度関数 (PDF) に対して反復の表現を得ることができる。

$$p(\lambda \mid \chi^n) = \frac{p(\mathbf{X}_n \mid \chi^{n-1}, \lambda) \cdot p(\lambda \mid \chi^{n-1})}{p(\mathbf{X}_n \mid \chi^{n-1})} \tag{2}$$

$$= \frac{p(\mathbf{X}_n \mid \lambda) \cdot p(\lambda \mid \chi^{n-1})}{\int_\Omega p(\mathbf{X}_n \mid \lambda) \cdot p(\lambda \mid \chi^{n-1}) d\lambda} \tag{3}$$

$p(\lambda \mid \chi^0) = p(\lambda)$ から PDF の計算を始めて、密度系列 $p(\lambda \mid \chi^1)$,$p(\lambda \mid \chi^2)$, $\cdots$ を生成する上式の使用を繰り返していく。

2.2 近似解:quasi-Bayes 学習法

反復 bayes 学習法のそれぞれのステップで、真の分布 $p(\lambda \mid \chi^n)$ を $g(\lambda \mid \varphi^{(n)})$ で近似する (図 2)。

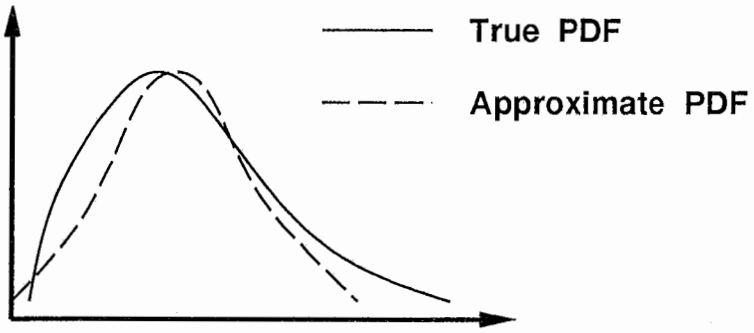

図 2: quasi-Bayes learning

$$p(\lambda \mid \chi^n) \rightarrow g(\lambda \mid \varphi^{(n)}) \tag{4}$$

ここに $\varphi^{(n)}$ は、サンプル $\mathbf{X}_n$ が出現した後にアップデートされたハイパーパラメータである。

$$\varphi^{(n)} = arg \min_{\varphi} \int p(\lambda \mid \chi^n) \log \frac{p(\lambda \mid \chi^n)}{g(\lambda \mid \varphi)} d\lambda \tag{5}$$

3 インプリメンテーション

インプリメントにあたっては、既存のSSS-ToolKitを利用することができるので、それを利用することを考える。

CDHMMパラメータのアップデートの式は次の通りである。

$$\hat{a}_{ij} = \frac{\rho\cdot(\eta_{ij}^{(n-1)}-1)+\sum_{all}\sum_{t=1}^{T_n}\gamma_t(i,j)}{\sum_{j=1}^{N}\rho\cdot(\eta_{ij}^{(n-1)}-1)+\sum_{j=1}^{N}\sum_{all}\sum_{t=1}^{T_n}\gamma_t(i,j)} \tag{6}$$

$$\hat{\omega}_{ik} = \frac{\rho\cdot(\nu_{ik}^{(n-1)}-1)+\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)}{\sum_{k=1}^{K}\rho\cdot(\nu_{ik}^{(n-1)}-1)+\sum_{k=1}^{K}\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)} \tag{7}$$

$$\hat{m}_{ikd} = \frac{\rho\tau_{ikd}^{(n-1)}\mu_{ikd}^{(n-1)}+\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)x_{td}}{\rho\tau_{ikd}^{(n-1)}+\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)} \tag{8}$$

$$\hat{r}_{ikd}^{-1} = \frac{2\rho\beta_{ikd}^{(n-1)}+\rho\tau_{ikd}^{(n-1)}(\hat{m}_{ikd}-\mu_{ikd}^{(n-1)})^2+\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)(x_{td}-\hat{m}_{ikd})^2}{\rho(2\alpha_{ikd}^{(n-1)}-1)+\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)} \tag{9}$$

ここに

$$\gamma_t(i,j) = Pr(s_t=i, s_{t+1}=j \mid \mathbf{X},\lambda)\ \ 1\le t\le T-1 \tag{10}$$

$$\zeta_t(i,k) = Pr(s_t=i, l_t=k \mid \mathbf{X},\lambda)\ \ 1\le t\le T \tag{11}$$

これらの項は前向き後向きアルゴリズムを用いて効率的に計算されうる。式中の$\sum_{all}$は、1発話中の全データに対する総和を意味する。SSS Toolkitにおけるi、jはステイトナンバーであり、例えば同じ発話中の同じ単語に関しては、同じステイトナンバーが対応する。従って、1発話中に同じ単語が出てくる場合(ex. もしもし)、それぞれについて、γ、ζを数えあげる必要がある。

最後の反復(M)においては、ハイパーパラメータも次のようにアップデートされる。

$$\eta_{ij}^{(n)} = \rho\cdot(\eta_{ij}^{(n-1)}-1)+1+\sum_{all}\sum_{t=1}^{T_n}\gamma_t(i,j) \tag{12}$$

$$\nu_{ik}^{(n)} = \rho\cdot(\nu_{ik}^{(n-1)}-1)+1+\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k) \tag{13}$$

$$\tau_{ikd}^{(n)} = \rho\tau_{ikd}^{(n-1)}+\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k) \tag{14}$$

$$\mu_{ikd}^{(n)} = \frac{\rho\tau_{ikd}^{(n-1)}\mu_{ikd}^{(n-1)}+c_{ik}\bar{x}_{ikd}}{\rho\tau_{ikd}^{(n-1)}+c_{ik}} \tag{15}$$

$$\alpha_{ikd}^{(n)} = \rho\cdot(\alpha_{ikd}^{(n-1)}-0.5)+0.5+\frac{1}{2}\sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k) \tag{16}$$

$$\beta_{ikd}^{(n)} = \rho\beta_{ikd}^{(n-1)}+\frac{1}{2}S_{ikd}+\frac{\rho\tau_{ikd}^{(n-1)}c_{ik}}{2(\rho\tau_{ikd}^{(n-1)}+c_{ik})}(\bar{x}_{ikd}-\mu_{ikd}^{(n-1)})^2 \tag{17}$$

ここに

$$c_{ik} = \sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k) \tag{18}$$

$$\bar{x}_{ikd} = \sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)x_{td}/c_{ik} \tag{19}$$

$$S_{ikd} = \sum_{all}\sum_{t=1}^{T_n}\zeta_t(i,k)(x_{td}-\bar{x}_{ikd})^2 \tag{20}$$

初期ハイパーパラメータの推定式は以下の通りである。

$$\eta_{ij}^{(0)} = 1 + \epsilon_1 \cdot \sum_t \gamma_t^{(SI)}(i,j) \tag{21}$$

$$\nu_{ik}^{(0)} = 1 + \epsilon_1 \cdot \sum_t \zeta_t^{(SI)}(i,k) \tag{22}$$

$$\tau_{ik}^{(0)} = \epsilon_1 \cdot \sum_t \zeta_t^{(SI)}(i,k) \tag{23}$$

$$\mu_{ik}^{(0)} = m_{ik}^{(SI)} \tag{24}$$

$$\alpha_{ikd}^{(0)} = \frac{1}{2} + \frac{1}{2}\epsilon_1 \cdot \sum_t \zeta_t^{(SI)}(i,k) \tag{25}$$

$$\beta_{ikd}^{(0)} = \frac{1}{2}\epsilon_1 \cdot (r_{ikd}^{(SI)})^{-1} \sum_t \zeta_t^{(SI)}(i,k) \tag{26}$$

ここに $0 < \epsilon_1 \leq 1$ は重み係数。

実際には、CDHMM パラメータのアップデートの式 (4) ～ (7) は、ハイパーパラメータを用いて次のように表されるので求める必要はないが、デバッグのためにこれら (4) ～ (7) も求めることとする。

$$\hat{a}_{ij} = \frac{\eta_{ij}^{(n)} - 1}{\sum_{j=1}^{N}(\eta_{ij}^{(n)} - 1)} \tag{27}$$

$$\hat{\omega}_{ik} = \frac{\nu_{ik}^{(n)} - 1}{\sum_{k=1}^{K}(\nu_{ik}^{(n)} - 1)} \tag{28}$$

$$\hat{m}_{ikd} = \mu_{ik}^{(n)} \tag{29}$$

$$\hat{r}_{ikd}^{-1} = \frac{\beta_{ikd}}{\alpha_{ikd} - 0.5} \tag{30}$$

以上のアルゴリズムの流れをわかりやすく示すと次のようになる。

```
initial HMM parameter: ahatij = aij
                       omagahatik = omegaik
                       mhatikd = mik
                       rhatikd = rik

initial hyperparameter: etahatij = etaij
                        nuhatik = nuik
                        tauhatik = tauik
                        muhatik = muik
                        alphahatikd = alphaikd
                        betahatikd = betaikd

for each utterance n

          for iteration m

                    gamma{i,j}( )
                    zeta{i,k}( )

                    update HMM parameter for debug: HMMomagahatik
                                                    HMMmhatikd
                                                    HMMmhatikd

                    update HMM parameter using hyperparameter: omegahatik(nuhatik)
                                                               mhatikd(muhatik)
                                                               rhatikd(alphahatikd,betahatikd)

                     for calculating hyperparameter: etaij( )
                                                     nuik( )
                                                     tauikd( )
                                                     muikd( )
                                                     alphaikd( )
                                                     betaikd( )

          end

          update hyperparameter: etahatij = etaij
                                 nuhatik = nuik
                                 tauhatik = tauik
                                 muhatik = muik
                                 alphahatikd = alphaikd
                                 betahatikd = betaikd

end
```

4　従来の適応法を用いた比較実験

これまでの話者適応法には、大きく分けて、既存の話者から得られた情報と適応話者の発話データから得られた情報を情報量に応じて統計的に結合するMAP推定法と、話者の特徴を表す音響的な空間は別の話者に連続的に移動できると仮定することにより不足した情報を獲得された情報をもとに補間・平滑化するVFS（移動ベクトル場平滑化法）の二つがある。

従来の適応法を用いた比較実験を行なった。

学習には3人の男性、3人の女性話者[3](MAU、MHT、FYM、MXM、FMS、FTK)によって話された最も頻繁に出てくる5240の日本語単語のうち、偶数番目のものを使用した。文脈依存(CD)、1300異音を表す400状態のHMMを生成するのには、逐次状態分割(SSS)アルゴリズム[6, 5]を用い、1状態の混合数3に対して、Baum-Welchのアルゴリズムを用いて再学習した。

認識は2人の男性、2人の女性話者(MNM、MSH、FFS、FKM)のフレーズ発話(サブセットSB3からの5、10、15、20、25番目のもの)に対して、ワンパスのViterbiアルゴリズムを用いて行なった。

適応は、4人のテスト話者のそれぞれに対して、サブセットSB1からの、2、4、6、8、10の適応文を用いて行なった。適応法としては、VFS[4]、standard MAP(MAP1)[1]、combined MAP and global VFS(MAP2)[7]、combined MAP and data-size dependent VFS(MAP3)[8]を使用した。スムージング・レイトは5.0、近傍数は6である。

図3は、それぞれの適応法による話者適応の結果である。認識結果は、インサーションやデリーションを含む、音素認識アキュラシーで示してある。各適応法におけるプロットは、それぞれ4人の話者による結果の平均である。適応文が増えるごとに、認識率が向上しているのがわかる。

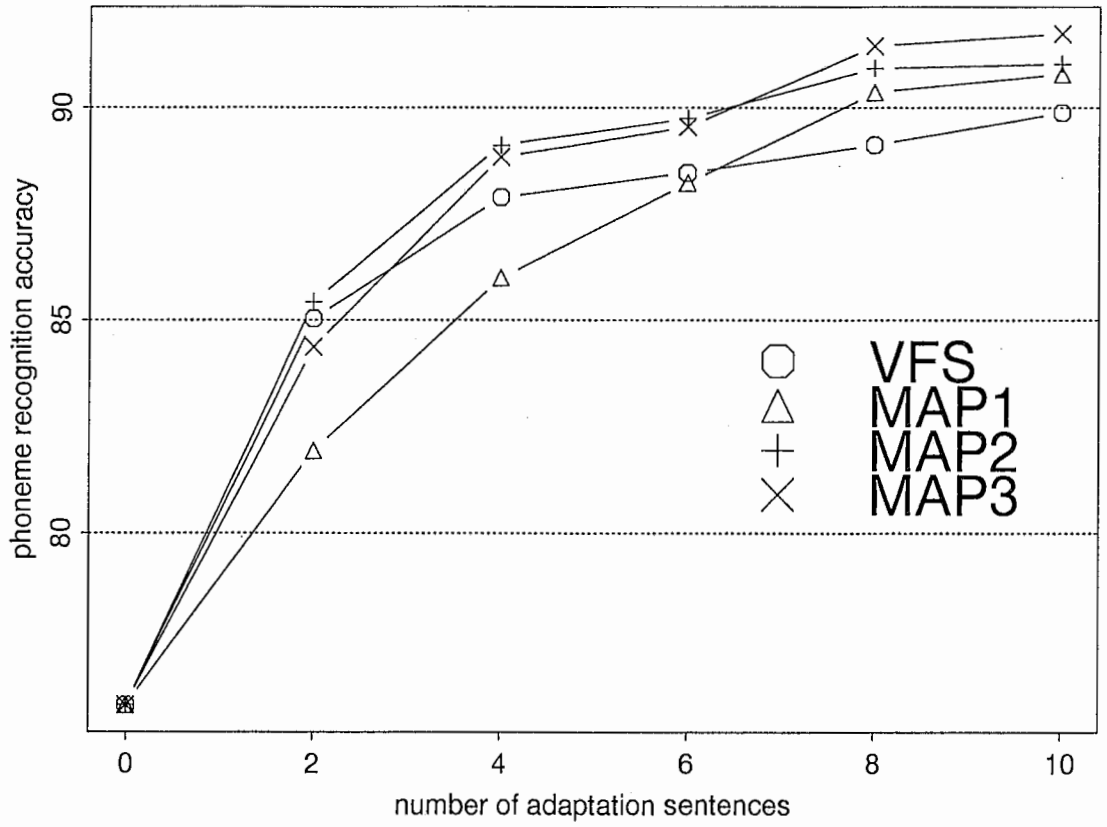

図 3: Baseline results (average)

4人の話者のそれぞれの結果を図4に示す。どの話者にしても、大体同じような傾向があることがわかる。最初の方ではMAPの結果の方がVFSに較べて悪いが、適応文が増すとVFSの方が悪くなってくる。これはVFSの

スムージングによるものであると考えられる。

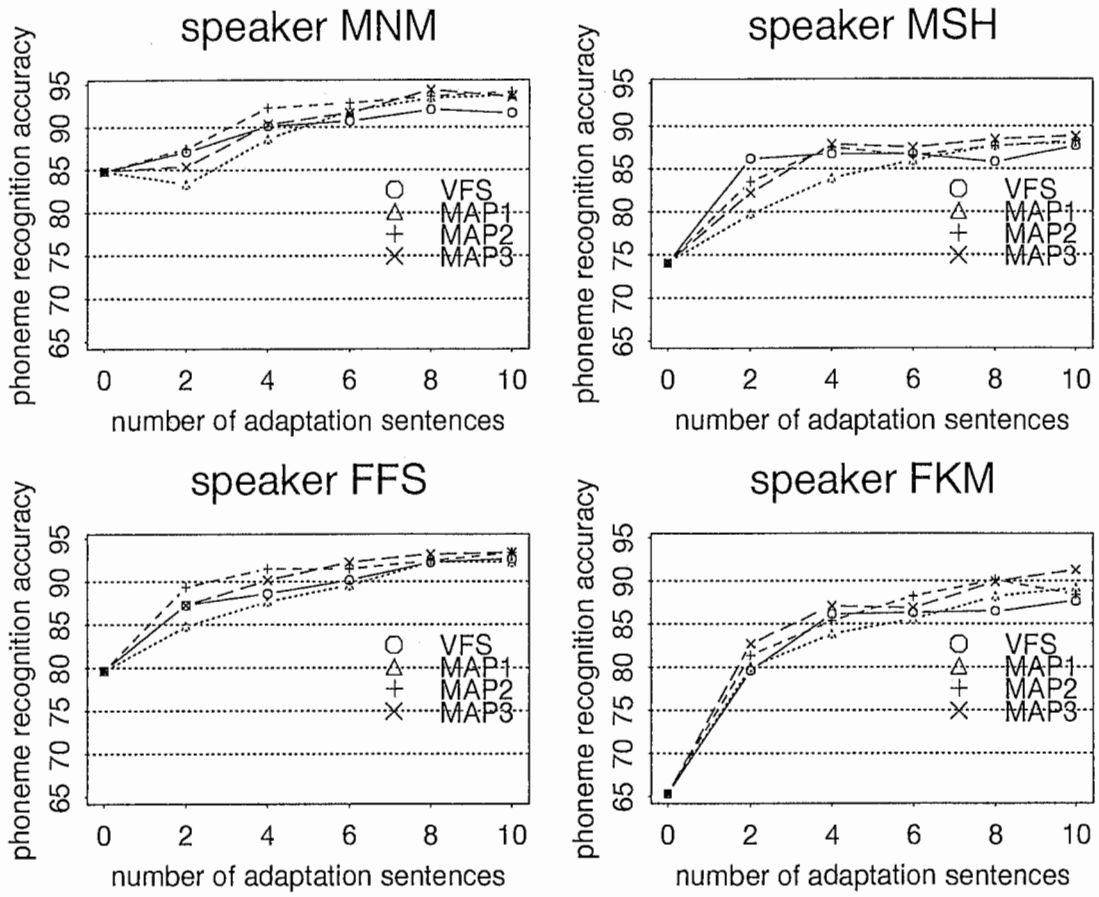

図 4: Baseline results (each speaker)

表 1 に適応データの詳細を示す。これは 4 人の中の 1 人の話者、FKM に関するデータである。適応文が増えるごとに、cputime が上昇していることがわかる。これは主に Baum-Welch のアルゴリズムにかかっている。従来の適応では、新たな適応文が追加されるごとに、全データを用いた再学習が必要であるので、このように cputime は無限に増加する。quasi-Bayes を用いれば、cputime はほぼ一定になることが予想される。これは quasi-Bayes が一つ前の適応データしか必要としないからである。

表 1: Example of adaptation data for speaker FKM

sentences	phrases	phonemes	frames	cputime (HP735 in s)
2	13	130	2143	72
4	35	349	5692	177
6	65	598	10085	287
8	77	741	12413	369
10	94	907	15252	448

5　まとめと今後の課題

今回は、実際のプログラミングにまで到達することができなかったため、今後はそのインプリメントとそれを用いた実験が望まれる。また従来のMAPやVFSでは、適応サンプルに新たな適応データを追加すると、全データによる再学習が必要となる。したがってそれらを用いた逐次適応を行なう場合、CPU TIMEは確実に上昇し、発話回数の増加につれてマシンに対する負担は増大する。quasi-Bayes学習法は一つ前の発話のみを用いて逐次適応を行なうため、CPU TIMEはほぼ一定値になることが予想される。もしもこの結果と同等、あるいはこれよりもずっと良い結果がこのquasi-Bayes学習法を用いて得られれば、マシンに対する負担の面から、ずっと効率的に話者適応を行なうことができると考えられる。今後はMAP、VFSといった従来の話者適応方式と、認識率、実行時間の面での性能の比較実験を行なうことが必要である。

6 謝辞

本実習の機会を与えて下さったATR音声翻訳通信研究所の山崎泰弘社長、並びにATR音声翻訳通信研究所第一研究室の匂坂芳典室長に深く感謝致します。また、快適な実習環境を提供して頂いたATR音声翻訳通信研究所第一研究室の研究員諸氏に感謝の意を表します。

参考文献

[1] J. Gauvain and C. Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Trans. Speech Audio*, 2(2):291–298, 1994.

[2] Q. Huo and C.H. Lee. On-line adaptive learning of the continuous density Hidden Markov Model based on approximate recursive Bayes estimate. *IEEE Trans. Speech Audio*, 1996. (submitted).

[3] A. Kurematsu, K. Takeda, Y. Sagisaka, S. Katagiri, H. Kuwabara, and K. Shikano. ATR Japanese speech database as a tool of speech recognition and synthesis. *Speech Communication*, 9:357–363, 1990.

[4] K. Ohkura, M. Sugiyama, and S. Sagayama. Speaker adaptation based on transfer vector field smoothing with continuous mixture density HMMs. In *Proc. ICSLP*, pages 369–372, Banff, 1992.

[5] M. Ostendorf and H. Singer. Modification of SSS for speaker-independent HM-net design. Technical Report TR-IT-0117, ATR, 1995.

[6] J. Takami and S. Sagayama. A successive state splitting algorithm for efficient allophone modeling. In *Proc. ICASSP*, volume 1, pages 573–576, San Francisco, March 1992.

[7] M. Tonomura, T. Kosaka, and S. Matsunaga. Speaker adaptation based on transfer vector field smoothing using maximum a posteriori probability estimation. In *Proc. ICASSP*, Detroit, 1995.

[8] M. Tonomura, T. Kosaka, S. Matsunaga, and A. Monden. Speaker adaptation fitting training data size and contents. In *Proc. EuroSpeech*, Madrid, 1995. (accepted).