**CAMBRIDGE UNIVERSITY**

ENGINEERING DEPARTMENT


# CLUSTER ADAPTIVE TRAINING OF HIDDEN MARKOV MODELS


M.J.F. Gales[1]

August 23, 1999

Cambridge University Engineering Department
Trumpington Street
Cambridge. CB2 1PZ
England

E-mail: `mjfg@eng.cam.ac.uk`
`http://www-svr.eng.cam.ac.uk/~mjfg`


[1]This work was performed whilst the author was with the IBM T.J. Watson Research Center speech group

**Abstract**

When performing speaker adaptation there are two conflicting requirements. First the transform must be powerful enough to represent the speaker. Second the transform must be quickly and easily estimated for any particular speaker. The most popular adaptation schemes have used many parameters to adapt the models to be representative of an individual speaker. This limits how rapidly the models may be adapted to a new speaker or acoustic environment. This paper examines an adaptation scheme requiring very few parameters, cluster adaptive training (CAT). CAT may be viewed as a simple extension to speaker clustering. Rather than selecting a single cluster as representative of a particular speaker, a linear interpolation of all the cluster means is used as the mean of the particular speaker. This scheme naturally falls into an adaptive training framework. Maximum likelihood estimates of the interpolation weights are given. Furthermore, simple re-estimation formulae for cluster means, represented both explicitly and by sets of transforms of some canonical mean, are given. On a speaker-independent task CAT reduced the word error rate using very little adaptation data. In addition when combined with other adaptation schemes it gave a 5% reduction in word error rate over adapting a speaker-independent model set.

# 1 Introduction

In recent years there has been a great deal of work done on adapting speech recognition systems to acoustic environment differences or to particular speakers [8, 12, 3, 17]. To adapt large numbers of parameters with very little adaptation data, a transform that modifies all the model parameters, not just those observed in the adaptation data, to be representative of a particular speaker or acoustic environment is required. The transform must be compact so that the parameters can be robustly estimated. However, the transform should also be powerful enough to accurately model the speaker or acoustic environment. A variety of transforms have been examined, for example, vocal tract normalisation [10], maximum likelihood linear regression (MLLR) [12], constrained model-space transforms [3, 7] and speaker clustering [16, 14]. The majority of these techniques apply some transformation to a *canonical model*. Originally a speaker-independent (SI) model was used as the canonical model. During recognition, the speaker-specific transform is applied to the SI model to generate the speaker-specific model. Recently *adaptive training*[2] was proposed as an alternative technique to generate the canonical model [1, 10]. Since the vast majority of training databases contain speech from many speakers, and in some cases acoustic environments, the adaptation scheme applied in recognition could also be used during training. By using adaptive training it is possible to build a canonical model which models the "average" variability for each of the training speakers rather than the variability over all speakers in the training database, as in the SI model. A currently popular version of adaptive training is speaker adaptive training (SAT) [1]. Here an MLLR transform is used to represent a speaker, both during training and testing. This particular form of adaptive training is related to the scheme described here. Another form of adaptive training used is speaker clustering [16, 14]. Here each of the training speakers is uniquely "assigned" to a speaker cluster. Then a model is built for each speaker cluster.

This paper considers a new, simple, form of adaptive training, *cluster adaptive training* (CAT)[3]. The approach is closely related to speaker clustering. However in contrast to standard speaker clustering, a linear interpolation of all the clusters parameters is used, rather than selecting a single cluster. To simplify the estimation process the form of the clusters is restricted slightly. The component priors and variances are tied over all the speaker clusters. Hence only the component means vary between clusters. For any particular speaker a set of interpolation values, the *cluster weight vector*, must be estimated. The mean of the Gaussian component for the speaker is then found by linearly combining the cluster means weighted according to the cluster weight vector. The form of the cluster means must also be chosen. This paper considers two types of clusters. The first is an explicit set of means per cluster. This will be referred to as *model-based* clusters. Alternatively, cluster-dependent transforms of some canonical means may be used for the cluster means, *transform-based* clusters. In both cases simple closed-form maximum likelihood (ML) estimation formulae are given. This new adaptation scheme allows very rapid adaptation, since only a small number of parameters, the cluster weight vector, needs to be estimated for a particular speaker. Furthermore, by using transform-based clusters, the increase in memory requirements for the model set may be made very small.

This paper is organised as follows. The next section describes the basic form of CAT and the overall training strategy. The following section describes how the cluster weight vectors may be estimated in an ML fashion. Estimation schemes for both model-based and transform-based clusters are then presented. Implementation issues, including possible schemes for initialising CAT, are described. A simple extension to SAT using a modified version of MLLR is then discussed. Results on a large-vocabulary task are presented and conclusions drawn.

# 2 Cluster Adaptive Training

CAT may be viewed as a simple extension to the standard speaker clustering scheme. Rather than assuming that the speaker belongs to one of a set of distinct speaker clusters, the speaker model's

[2] Authors have also used the term *adapted training*.

[3] A shortened version of this paper describing the basic CAT algorithm was presented at ICSLP'98 [6].

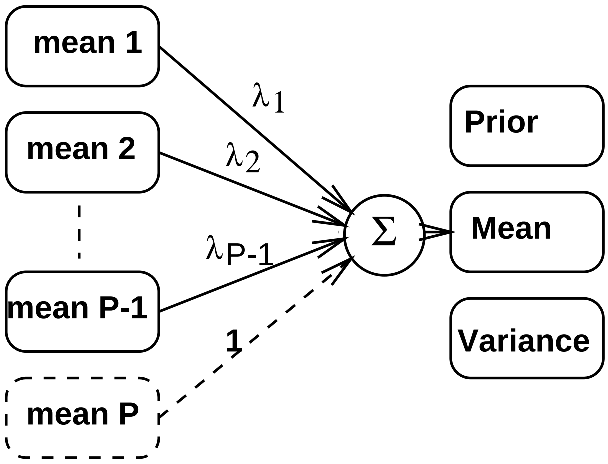

Figure 1: Cluster adaptive training

mean parameters are determined by a linear combination of all the cluster means. The Gaussian component variances and prior are assumed to be the same across all clusters and independent of the particular speaker.

CAT is shown diagrammatically in figure 1. The $P$ sets of cluster means are linearly combined using a set of interpolation weights, the cluster weight vector, $\boldsymbol{\lambda} = \left[ \begin{array}{cccc} \lambda_1 & \dots & \lambda_{P-1} & 1 \end{array} \right]$. Figure 1 also shows a *bias cluster*. This is indicated using a dotted line. The bias cluster is used to represent any speaker independent aspects of the mean vectors, given the particular representation of the speaker. It always has a weight of 1. Further discussions of the possible use of a bias cluster are given in section 6. It is not necessary to use a single cluster weight vector for each speaker. To add flexibility to the system multiple cluster weight vectors may be used in CAT in a similar fashion to the piecewise linear approximation used in MLLR [11]. The Gaussian components are partitioned into a set of $R$ disjoint cluster weight classes, $M_{\boldsymbol{w}}^{(1)}$ to $M_{\boldsymbol{w}}^{(R)}$. Each cluster weight class has a separate cluster weight vector calculated for it, which is used to adapt the Gaussian components belonging to that class. There are a variety of methods for partitioning the Gaussian components, similar to the partitions for MLLR discussed in [4]. Standard schemes for partitioning the components include clustering in acoustic space and using phonetic questions. Using multiple cluster weight vectors increases the number of parameters to be estimated prior to recognition, but makes the system more flexible. The number of cluster weight vectors may be dynamically chosen depending on the available adaptation data using, for example, a regression class tree [11].

CAT is defined as follows. For a particular Gaussian component, $m \in M_{\boldsymbol{w}}^{(r_m)}$, the mean for speaker $s$, $\boldsymbol{\mu}^{(sm)}$, is given by

$$\boldsymbol{\mu}^{(sm)} = \boldsymbol{M}^{(m)} \boldsymbol{\lambda}^{(sr_m)} \tag{1}$$

where $\boldsymbol{M}^{(m)}$ is the matrix of $P$ cluster mean vectors for component $m$,[4]

$$\boldsymbol{M}^{(m)} = \left[ \begin{array}{ccc} \boldsymbol{\mu}_{\boldsymbol{c}}^{(m1)} & \dots & \boldsymbol{\mu}_{\boldsymbol{c}}^{(mP)} \end{array} \right] \tag{2}$$

where $\boldsymbol{\mu}_{\boldsymbol{c}}^{(mp)}$ is the mean of Gaussian component $m$ associated with cluster $p$ and the cluster weight vector for speaker $s$, $\boldsymbol{\lambda}^{(sr_m)}$, is given by (assuming that a bias cluster is used)

$$\boldsymbol{\lambda}^{(sr_m)} = \left[ \begin{array}{cccc} \lambda_1^{(sr_m)} & \dots & \lambda_{P-1}^{(sr_m)} & 1 \end{array} \right]^T \tag{3}$$

[4]This paper uses the following convention: capital bold letters refer to matrices e.g. $\boldsymbol{A}$, bold letters refer to vectors e.g. $\boldsymbol{b}$, and scalars are not bold e.g. $c$. When referring to elements of a matrix or vector subscripts are used e.g. $\boldsymbol{a}_i$ is the $i^{th}$ row of matrix $\boldsymbol{A}$, $a_{ij}$ is the element of row $i$ column $j$ of matrix $\boldsymbol{A}$ and $b_i$ is element $i$ of vector $\boldsymbol{b}$. Bold subscripts are used to discriminate between different, but related variables e.g. $\boldsymbol{A}_{\boldsymbol{t}}$ and $\boldsymbol{A}_{\boldsymbol{f}}$.

and $r_m$ is the cluster weight class of Gaussian component $m$.

The ML training scheme for CAT is similar to other adaptive training schemes [1]. In CAT, the canonical model, $\mathcal{M}$, consists of[5]

$$\mathcal{M} = \left\{ \{\boldsymbol{M}^{(1)}, \ldots, \boldsymbol{M}^{(M)}\}, \{\boldsymbol{\Sigma}^{(1)}, \ldots, \boldsymbol{\Sigma}^{(M)}\} \right\} \tag{4}$$

where $M$ is the total number of Gaussian components and $\boldsymbol{\Sigma}^{(m)}$ is the covariance matrix associated with Gaussian component $m$. To find the ML estimate of the canonical model parameters, the set of ML cluster weight vectors for all the training speakers must also be found. The aim of the training scheme is to maximise the likelihood of the training data $\boldsymbol{O}^{(T)}$, $\boldsymbol{O}^{(T)} = \boldsymbol{o}(1), \ldots, \boldsymbol{o}(T)$ with associated transcriptions and speaker labels, i.e. maximise

$$\mathcal{L}(\boldsymbol{O}^{(T)}; \mathcal{M}, \boldsymbol{\Lambda}) = \prod_{s} \sum_{\boldsymbol{\Theta}^{(s)}} \prod_{\tau} \left( p(q(\tau)|q(\tau-1)) \sum_{m \in \boldsymbol{\theta}(\tau)} w^{(m)} \mathcal{N}(\boldsymbol{o}(\tau); \boldsymbol{M}^{(m)} \boldsymbol{\lambda}^{(sr_m)}, \boldsymbol{\Sigma}^{(m)}) \right) \tag{5}$$

where $\boldsymbol{\Theta}^{(s)}$ is the set of all valid state sequences according to the transcription for the data from speaker $s$, $q(\tau)$ is the state at time $\tau$ of the current path and $\boldsymbol{\theta}(\tau)$ is set of Gaussian components belonging to state $q(\tau)$, $w^{(m)}$ is the prior of Gaussian component $m$ and $\mathcal{N}(\boldsymbol{o}(\tau); \boldsymbol{\mu}^{(sm)}, \boldsymbol{\Sigma}^{(m)})$ is the likelihood of the n-dimensional observation vector at time $\tau$, $\boldsymbol{o}(\tau)$, from Gaussian component $m$, having mean $\boldsymbol{\mu}^{(sm)}$ and covariance matrix $\boldsymbol{\Sigma}^{(m)}$, with respect to the canonical model parameters, $\mathcal{M}$, and the set of $S$ training speaker cluster weight vectors, $\boldsymbol{\Lambda}$,

$$\boldsymbol{\Lambda} = \left\{ \{\boldsymbol{\lambda}^{(11)}, \ldots, \boldsymbol{\lambda}^{(1R)}\}, \ldots, \{\boldsymbol{\lambda}^{(S1)}, \ldots, \boldsymbol{\lambda}^{(SR)}\} \right\} \tag{6}$$

Directly optimising equation 5 is a very high dimensional problem. There may be tens of thousands of Gaussian components and thousands of training speakers. In a similar fashion to standard HMM training, expectation-maximisation (EM) [2] may be used to estimate the parameters. EM uses an "old" set of model parameters to estimate a *complete dataset*[6] and then estimates "new" model parameters using this complete dataset. The auxiliary function to be maximised with respect to the new parameters $\mathcal{M}$ and $\boldsymbol{\Lambda}$ is (ignoring the standard constants and terms independent of $\mathcal{M}$ and $\boldsymbol{\Lambda}$)

$$\mathcal{Q}(\mathcal{M}, \boldsymbol{\Lambda}; \hat{\mathcal{M}}, \hat{\boldsymbol{\Lambda}}) = -\frac{1}{2} \sum_{s,m,\tau} \gamma_m(\tau) \left( \boldsymbol{o}(\tau) - \boldsymbol{M}^{(m)} \boldsymbol{\lambda}^{(sr_m)} \right)^T \boldsymbol{\Sigma}^{(m)-1} \left( \boldsymbol{o}(\tau) - \boldsymbol{M}^{(m)} \boldsymbol{\lambda}^{(sr_m)} \right) \tag{7}$$

where $\gamma_m(\tau)$ is the posterior probability of Gaussian component $m$ at time $\tau$ determined using the old model parameters, $\hat{\mathcal{M}}$ and $\hat{\boldsymbol{\Lambda}}$. There is no simple solution to simultaneously estimating both $\mathcal{M}$ and $\boldsymbol{\Lambda}$. Hence an iterative training scheme is used, where either the canonical model, $\mathcal{M}$, or the speaker cluster weight vectors, $\boldsymbol{\Lambda}$, are updated. Thus the scheme is:

1. Estimate the values of the cluster weight vectors for each speaker in the training data, given the current estimate of the canonical model parameters and old cluster weight vectors, which are used to obtain the complete dataset.

2. Estimate a new canonical model given the new cluster weight vectors and the old canonical model, again used to estimate a new complete dataset.

3. Go to (1) until the convergence criterion satisfied.

Each of these optimisation stages is described in detail in the next two sections.

[5] Here the set of Gaussian component priors, or weights, $\{w^{(1)}, \ldots, w^{(M)}\}$ , and the state transition probabilities, have not been included. The estimation of these parameters, as will be seen is very similar to the standard weight estimation scheme.

[6] Here the complete dataset consists of the observations $\boldsymbol{o}(1), \ldots, \boldsymbol{o}(T)$ and the posterior distribution of the observation at time $\tau$ being generated by Gaussian component $m$, $q_m(\tau)$.

## 3 Cluster Weight Vectors

The ML estimation of the cluster weight vectors is similar to the training of the smoothing weights in transformation smoothing described in [4]. Initially consider the case where no bias cluster is being used. Equation 7 may be rewritten, ignoring terms independent of the cluster weight vector and fixing the canonical model parameters, $\mathcal{M}$, as

$$\mathcal{Q}(\boldsymbol{\Lambda};\mathcal{M},\hat{\boldsymbol{\Lambda}}) = \sum_{s,r} \boldsymbol{\lambda}^{(sr)}\boldsymbol{k}_{\boldsymbol{w}}^{(sr)} - \frac{1}{2}\boldsymbol{\lambda}^{(sr)T}\boldsymbol{G}_{\boldsymbol{w}}^{(sr)}\boldsymbol{\lambda}^{(sr)} \tag{8}$$

where the accumulates for cluster weight class $r$ of each speaker $s$, $\boldsymbol{G}_{\boldsymbol{w}}^{(sr)}$ and $\boldsymbol{k}_{\boldsymbol{w}}^{(sr)}$, are given by

$$\boldsymbol{G}_{\boldsymbol{w}}^{(sr)} = \sum_{m\in M_{\boldsymbol{w}}^{(r)},\tau} \gamma_m(\tau)\boldsymbol{M}^{(m)T}\boldsymbol{\Sigma}^{(m)-1}\boldsymbol{M}^{(m)} \tag{9}$$

$$\boldsymbol{k}_{\boldsymbol{w}}^{(sr)} = \sum_{m\in M_{\boldsymbol{w}}^{(r)}} \boldsymbol{M}^{(m)T}\boldsymbol{\Sigma}^{(m)-1}\sum_{\tau}\gamma_m(\tau)\boldsymbol{o}(\tau) \tag{10}$$

and $\gamma_m(\tau)$ is determined using the canonical model parameters $\mathcal{M}$ and the old cluster weight vectors $\hat{\boldsymbol{\Lambda}}$. By differentiating with respect to the cluster weight vector of a particular speaker and equating to zero, the cluster weight vector for speaker $s$ associated with cluster weight class $r$ may be shown to be

$$\boldsymbol{\lambda}^{(sr)} = \boldsymbol{G}_{\boldsymbol{w}}^{(sr)-1}\boldsymbol{k}_{\boldsymbol{w}}^{(sr)} \tag{11}$$

Only $P-1$ cluster weight vector components must be estimated if a bias cluster is used, since the bias cluster interpolation weight has a fixed value. A similar estimation scheme is used, except that equation 10 is replaced by (here the $P^{th}$ cluster is the bias cluster)

$$\boldsymbol{k}_{\boldsymbol{w}}^{(sr)} = \sum_{m\in M_{\boldsymbol{w}}^{(r)}} \boldsymbol{M}^{(m)T}\boldsymbol{\Sigma}^{(m)-1}\sum_{\tau}\gamma_m(\tau)(\boldsymbol{o}(\tau)-\boldsymbol{\mu}_{\boldsymbol{c}}^{(mP)}) \tag{12}$$

The sufficient statistics for estimating the cluster weight vector are highly compact. Only $\boldsymbol{G}_{\boldsymbol{w}}^{(sr)}$ and $\boldsymbol{k}_{\boldsymbol{w}}^{(sr)}$ are stored for each of the $R$ cluster weight classes[7]. For $P$ clusters the sufficient statistics require storing $P^2+P$ floats.

The standard "hard" speaker cluster, where each speaker is assigned to a particular cluster, may also be simply expressed in terms of $\boldsymbol{G}_{\boldsymbol{w}}^{(sr)}$ and $\boldsymbol{k}_{\boldsymbol{w}}^{(sr)}$. Now the ML cluster for cluster weight class $r$ of speaker $s$, $p^{(sr)\prime}$, is

$$p^{(sr)\prime} = \arg\max_{p}\left\{k_{\boldsymbol{w}p}^{(sr)} - \frac{1}{2}g_{\boldsymbol{w}pp}^{(sr)}\right\} \tag{13}$$

The equivalent cluster weight vector is given by

$$\lambda_p^{(sr)} = \begin{cases} 1, & p = p^{(sr)\prime} \\ 0, & \text{(otherwise)} \end{cases} \tag{14}$$

When this hard clustering is linked with the model-based clusters described in the next section and a single cluster weight class is used, it is identical to standard speaker clustering with the constraint that the Gaussian component priors and variances are tied over all clusters.

[7] For efficiency reasons, it may be preferable to store the Gaussian component occupancy and vector sum and then use these accumulates to form $\boldsymbol{G}_{\boldsymbol{w}}^{(sr)}$ and $\boldsymbol{k}_{\boldsymbol{w}}^{(sr)}$ when the cluster weight vectors are to be estimated.

# 4 Cluster Parameters

This section describes the estimation scheme for both the cluster means and the cluster independent Gaussian component priors and variances. Two representations of the cluster means are described. The first is an explicit set of means for each cluster, *model-based clusters*, as depicted in figure 1. The second representation is a cluster dependent linear transformation of a set of canonical means, *transform-based clusters*. Estimation of the transition probabilities in CAT is the same as standard HMM transition probability training irrespective of the form of the cluster and is not further discussed in this paper.

The optimisation schemes for model-based and transform-based clusters are given below. Both aim to maximise the following auxiliary function

$$\mathcal{Q}(\mathcal{M};\hat{\mathcal{M}},\boldsymbol{\Lambda}) = -\frac{1}{2}\sum_{s,m,\tau}\gamma_m(\tau)\left(\boldsymbol{o}(\tau)-\boldsymbol{M}^{(m)}\boldsymbol{\lambda}^{(sr_m)}\right)^T\boldsymbol{\Sigma}^{(m)-1}\left(\boldsymbol{o}(\tau)-\boldsymbol{M}^{(m)}\boldsymbol{\lambda}^{(sr_m)}\right) \tag{15}$$

with respect to the canonical model, $\mathcal{M}$, where $\gamma_m(\tau)$ is the posterior probability of Gaussian component $m$ at time $\tau$ determined using the old canonical model parameters $\hat{\mathcal{M}}$ and the cluster weight vectors $\boldsymbol{\Lambda}$. The difference between the two types of cluster representation is the form of the matrix of cluster means, $\boldsymbol{M}^{(m)}$. The sufficient statistics (and the notation used for those statistics) for both cluster representations are described in the following section.

## 4.1 Statistics required

For both the model-based and transform-based representations of the cluster, the same sufficient statistics are required to estimate the cluster parameters. These are

$$\boldsymbol{G}^{(m)} = \sum_{s,\tau}\gamma_m(\tau)\boldsymbol{\lambda}^{(sr_m)}\boldsymbol{\lambda}^{(sr_m)T} \tag{16}$$

$$\boldsymbol{K}^{(m)} = \sum_{s,\tau}\gamma_m(\tau)\boldsymbol{\lambda}^{(sr_m)}\boldsymbol{o}(\tau)^T \tag{17}$$

$$\boldsymbol{L}^{(m)} = \sum_{s,\tau}\gamma_m(\tau)\boldsymbol{o}(\tau)\boldsymbol{o}(\tau)^T \tag{18}$$

and the number of frames assigned to that particular Gaussian component, $\sum_{s,\tau}\gamma_m(\tau)$. If diagonal covariance matrices are to be used, then it is only necessary to store $\mathrm{diag}\left(\boldsymbol{L}^{(m)}\right)$. Diagonal covariance will be assumed for the rest of this paper. The total memory requirement for the cluster estimation for an $n$-dimensional feature vector, $M$ Gaussian components, and $P$ clusters is $M(P^2+Pn+n+1)$ floats. Further discussion of the memory and computational cost of CAT is given in section 5.1.

The sufficient statistics for CAT training may be contrasted with those for speaker adapted training (SAT) [1]. In SAT there are no sufficient statistics of a fixed dimensionality. The statistics to be stored are a function of the number of speakers in the training data[8]. This is a not an important issue for small training sets, however when the number of speaker/noise conditions becomes large, for example on the ARPA Hub4 task where there are greater than 5,000 such conditions, standard SAT training rapidly becomes impractical. While there are possible solutions to this problem [13, 15], they typically significantly increase the training time, or are no longer based on an ML training criterion.

[8]This is not true of the modified SAT scheme described in [7].

## 4.2 Model-based clusters

With model-based clusters there is a mean estimate for each component of each cluster. Thus each element of the canonical model, $\mathcal{M}$, must be explicitly estimated. The adapted mean of Gaussian component $m$ of a particular speaker, $s$, using model-based clusters is given by equation 1. The problem is to find the matrix of cluster means, $\boldsymbol{M}^{(m)}$ (defined in equation 2). The estimation of the cluster means is a simplification of the soft MLLR transform estimation scheme described in [5]. In appendix A.1 it is shown that the set of mean vectors may be estimated using[9]

$$\boldsymbol{M}^{(m)T} = \boldsymbol{G}^{(m)-1}\boldsymbol{K}^{(m)} \tag{19}$$

where $\boldsymbol{G}^{(m)}$ and $\boldsymbol{K}^{(m)}$ are defined in equations 16 and 17. The variance is shown to be

$$\boldsymbol{\Sigma}^{(m)} = \text{diag}\left(\frac{\boldsymbol{L}^{(m)} - \boldsymbol{M}^{(m)}\boldsymbol{K}^{(m)}}{\sum_{s,\tau}\gamma_m(\tau)}\right) \tag{20}$$

## 4.3 Transform-based clusters

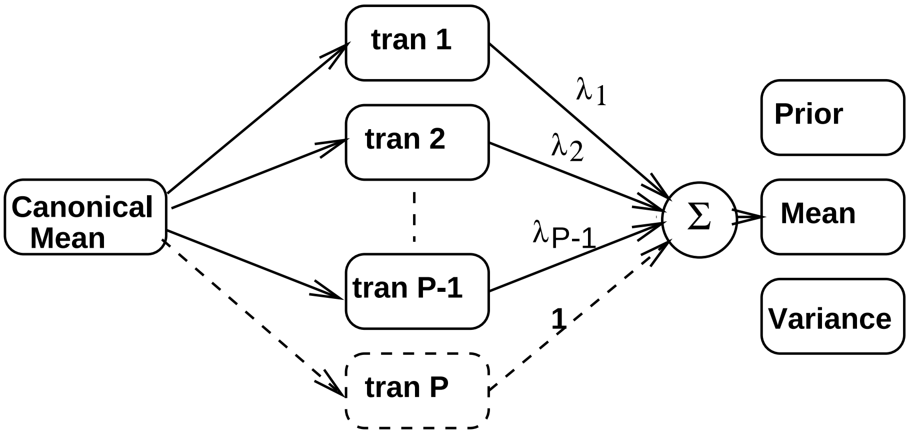


Figure 2: Transform-based cluster adaptive training

Figure 2 shows the structure of CAT with transform-based clusters. Rather than the clusters being represented as a distinct set of means, they are now a set of cluster-specific transforms of a set of canonical means. A bias cluster, indicated with a dotted line, is also shown. This bias cluster is shown as a linear transform of the set of canonical means, which will be the form considered in this paper. With slight modifications to the training algorithm the bias cluster may be represented as a separate set of distinct means.

The cluster-specific transform used in this work is MLLR [12][10].In common with many applications of MLLR a piecewise linear transformation of the canonical means will be used to represent each cluster. Thus the Gaussian components are partitioned into $Q$ transform classes, $M_{\boldsymbol{t}}^{(1)}$ to $M_{\boldsymbol{t}}^{(Q)}$. The transform associated with a particular transform class $q$ of a cluster $p$ is $\boldsymbol{W}_{\boldsymbol{c}}^{(pq)}$, where $\boldsymbol{W}_{\boldsymbol{c}}^{(pq)} = \begin{bmatrix} \boldsymbol{A}_{\boldsymbol{c}}^{(pq)} & \boldsymbol{b}_{\boldsymbol{c}}^{(pq)} \end{bmatrix}$. The mean of Gaussian component $m$ of cluster $p$ is given by

$$\boldsymbol{\mu}_{\boldsymbol{c}}^{(mp)} = \boldsymbol{A}_{\boldsymbol{c}}^{(pt_m)}\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} + \boldsymbol{b}_{\boldsymbol{c}}^{(pt_m)} = \boldsymbol{W}_{\boldsymbol{c}}^{(pt_m)}\boldsymbol{\xi}_{\boldsymbol{a}}^{(m)} \tag{21}$$

[9]The cluster means of a particular component are all estimated simultaneously. It is therefore simple in multiple Gaussian component states to associate Gausian components in the clusters together.

[10]Other transforms may be used to represent the cluster. In the general case transformations of all the model parameters are possible, i.e. the component priors, means and variances. However if the variances are adapted then the simple smoothing described here is not possible, though the hard clustering scheme may be used.

where $\boldsymbol{\xi}_{\boldsymbol{a}}^{(m)}$ is the extended mean vector of the canonical mean, $\left[\begin{array}{cc} \boldsymbol{\mu}_{\boldsymbol{a}}^{(m)T} & 1 \end{array}\right]^{T}$, $\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)}$ is the canonical mean of Gaussian component $m$ and $t_m$ is the transform class of Gaussian component $m$, i.e. $m \in M_{\boldsymbol{t}}^{(t_m)}$. The adapted mean of Gaussian component $m$ for a particular speaker $s$ is

$$\boldsymbol{\mu}^{(sm)} = \left( \sum_p \lambda_p^{(sr_m)} \boldsymbol{W}_{\boldsymbol{c}}^{(pt_m)} \right) \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)} \tag{22}$$

In transform-based CAT, the canonical model has two distinct elements. The first is the set of cluster dependent transforms, $\mathcal{T}$, where

$$\mathcal{T} = \left\{ \{\boldsymbol{W}_{\boldsymbol{c}}^{(11)}, \ldots, \boldsymbol{W}_{\boldsymbol{c}}^{(1Q)}\}, \ldots, \{\boldsymbol{W}_{\boldsymbol{c}}^{(P1)}, \ldots, \boldsymbol{W}_{\boldsymbol{c}}^{(PQ)}\} \right\} \tag{23}$$

The second element of the canonical model is the underlying set of Gaussian components to be modified[11], $\tilde{\mathcal{M}}$, where

$$\tilde{\mathcal{M}} = \left\{ \{\boldsymbol{\mu}_{\boldsymbol{a}}^{(1)}, \ldots, \boldsymbol{\mu}_{\boldsymbol{a}}^{(M)}\}, \{\boldsymbol{\Sigma}^{(1)}, \ldots, \boldsymbol{\Sigma}^{(M)}\} \right\} \tag{24}$$

Thus the canonical model used for transform-based cat is $\{\mathcal{T}, \tilde{\mathcal{M}}\}$.

In a similar fashion to standard SAT [1] an iterative estimation scheme is required to estimate $\mathcal{T}$ and $\tilde{\mathcal{M}}$, since there is no simple estimation scheme for simultaneously estimating both sets of parameters. Given some initial estimate of the canonical model and the cluster weight vector associated with each speaker, the training process is as follows.

1. Estimate the set of MLLR transforms, $\mathcal{T}$, given the model parameters, $\tilde{\mathcal{M}}$ and the training speaker cluster vectors, $\boldsymbol{\Lambda}$.
2. Estimate the model parameters, $\tilde{\mathcal{M}}$, given the current set of transforms, $\mathcal{T}$ and the set of cluster vectors, $\boldsymbol{\Lambda}$.
3. Go to (1) until the convergence criterion satisfied.

Each of these estimation steps will be described below.

CAT using transform-based clusters is closely related to SAT [1]. In contrast to SAT a fixed number of training "speakers", or clusters, are chosen. Using a fixed, usually small, number of clusters has some advantages over considering each training speaker independently. First, the re-estimation formulae become far easier as the statistics may be stored at the cluster level. Second, the amount of data associated with each cluster is far greater than that associated with the average training speaker. It is therefore possible to use many transforms to represent the cluster. Furthermore during recognition using CAT adaptation only interpolation weights for a set of transforms associated with the clusters are estimated for a new speaker, rather than a new estimate of a transform as in SAT. This significantly reduces the number of parameters to be estimated for a particular speaker. However the flexibility of CAT is restricted compared to SAT since only an interpolation of a fixed number of transforms is permitted.

**MLLR transforms.** The estimation of the transformation parameters is similar to the linear transform optimisation described in [4, 5]. In [5] it is shown that the parameters associated with row $i$ of transform class $q$ for all $P$ clusters may be simultaneously estimated using

$$\boldsymbol{H}^{(qi)} = \boldsymbol{G}_{\boldsymbol{f}}^{(qi)-1} \boldsymbol{k}_{\boldsymbol{f}}^{(qi)} \tag{25}$$

where

$$\boldsymbol{H}^{(qi)} = \left[ \begin{array}{ccc} \boldsymbol{w}_{\boldsymbol{c}i}^{(1q)} & \ldots & \boldsymbol{w}_{\boldsymbol{c}i}^{(Pq)} \end{array} \right]^{T} \tag{26}$$

[11] For consistency with the previous canonical model definition the Gaussian component priors $\{w^{(1)}, \ldots, w^{(M)}\}$ are not included.

and the accumulates for each transform class $q$ of each dimension $i$, $\boldsymbol{k}_{\boldsymbol{f}}^{(qi)}$ and $\boldsymbol{G}_{\boldsymbol{f}}^{(qi)}$, are functions of $\boldsymbol{G}^{(m)}$ and $\boldsymbol{K}^{(m)}$,

$$\boldsymbol{k}_{\boldsymbol{f}}^{(qi)} = \sum_{s,m\in M_{\boldsymbol{t}}^{(q)},\tau} \left( \boldsymbol{\lambda}^{(sr_m)T} \otimes \gamma_m(\tau) \frac{1}{\sigma_{ii}^{(m)2}} o_i(\tau) \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)T} \right) \tag{27}$$

$$\boldsymbol{G}_{\boldsymbol{f}}^{(qi)} = \sum_{s,m\in M_{\boldsymbol{t}}^{(q)},\tau} \left( \boldsymbol{\lambda}^{(sr_m)} \boldsymbol{\lambda}^{(sr_m)T} \otimes \gamma_m(\tau) \frac{1}{\sigma_{ii}^{(m)2}} \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)} \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)T} \right) \tag{28}$$

where $\otimes$ is the Kronecker product. However, this calculation involves the inversion of a large $P(n+1)$ square matrix, $n$ is the dimensionality of the feature vector. A simple efficient iterative approach is possible [5] if it is believed that the cluster weight vectors tend to be dominated by a single element. In practice, though this is not strictly true, this form of iterative approach was found to rapidly converge. Consider only one of the clusters, $p$. It is simple to estimate the transform parameters of cluster $p$ by keeping all other cluster transforms fixed. Row $i$ of the transform associated with transform class $q$ of cluster $p$ is found to be

$$\boldsymbol{w}_{\boldsymbol{c}i}^{(pq)} = \boldsymbol{k}_{\boldsymbol{t}}^{(pqi)} \boldsymbol{G}_{\boldsymbol{t}}^{(pqi)-1} \tag{29}$$

where the accumulates for transform class $q$ of cluster $p$ dimension $i$, $\boldsymbol{G}_{\boldsymbol{t}}^{(pqi)}$ and $\boldsymbol{k}_{\boldsymbol{t}}^{(pqi)}$, are again functions of $\boldsymbol{G}^{(m)}$ and $\boldsymbol{K}^{(m)}$

$$\boldsymbol{G}_{\boldsymbol{t}}^{(pqi)} = \sum_{m\in M_{\boldsymbol{t}}^{(q)}} g_{pp}^{(m)} \frac{1}{\sigma_{ii}^{(m)2}} \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)} \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)T} \tag{30}$$

$$\boldsymbol{k}_{\boldsymbol{t}}^{(pqi)} = \sum_{m\in M_{\boldsymbol{t}}^{(q)}} \frac{1}{\sigma_{ii}^{(m)2}} \left( k_{pi}^{(m)} - \sum_{j\neq p} g_{jp}^{(m)} \boldsymbol{w}_{\boldsymbol{c}i}^{(jq)} \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)} \right) \boldsymbol{\xi}_{\boldsymbol{a}}^{(m)T} \tag{31}$$

Since $\boldsymbol{G}_{\boldsymbol{t}}^{(pqi)}$ is independent of the current estimate of the transform, it is only estimated and inverted once. However, $\boldsymbol{k}_{\boldsymbol{t}}^{(pqi)}$ is dependent on the value of the other transforms, so must be updated as each transform changes. Having updated the transforms for one cluster, those of the next cluster may be estimated and so on. The process is quite efficient and typically converges after only a few iterations.

**Canonical Model Estimation.** There is now the issue of estimating the model parameters $\tilde{\mathcal{M}}$ which are transformed by the cluster dependent transforms, $\mathcal{T}$. The estimation of the means is slightly more complex than the model-based mean estimation and is closely linked to the SAT re-estimation formulae. The details of the derivation of the estimation formulae used for CAT are given in appendix A.2. The ML estimate of the mean of Gaussian component $m$ is shown to be

$$\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} = \left( \sum_{i,j} g_{ij}^{(m)} \boldsymbol{A}_{\boldsymbol{c}}^{(it_m)T} \boldsymbol{\Sigma}^{(m)-1} \boldsymbol{A}_{\boldsymbol{c}}^{(jt_m)} \right)^{-1} \sum_i \boldsymbol{A}_{\boldsymbol{c}}^{(it_m)T} \boldsymbol{\Sigma}^{(m)-1} \left( \boldsymbol{k}_i^{(m)T} - \sum_j g_{ij}^{(m)} \boldsymbol{b}_{\boldsymbol{c}}^{(it_m)} \right) \tag{32}$$

Similarly the variance may be expressed as

$$\boldsymbol{\Sigma}^{(m)} = \operatorname{diag}\left( \frac{\boldsymbol{L}^{(m)} - 2\sum_i \boldsymbol{k}_i^{(m)} \boldsymbol{\mu}_{\boldsymbol{c}}^{(mi)T} + \sum_{i,j} g_{ij}^{(m)} \boldsymbol{\mu}_{\boldsymbol{c}}^{(mi)} \boldsymbol{\mu}_{\boldsymbol{c}}^{(mj)T}}{\sum_{s,\tau} \gamma_m(\tau)} \right) \tag{33}$$

The estimation of the mean is a function of the variance and vice-versa. The optimisation of the parameters is thus an iterative procedure, optimising the means and then the variances and repeating.

# 5 Implementation Issues

## 5.1 Memory and computational costs

The two cluster representations have very different memory requirements and adaptation computational costs. Consider a system with $P$ clusters, $M$ Gaussian components and an $n$-dimensional feature vector. For the transform-based clusters $Q$ full MLLR transforms will be used for each cluster.

| Cluster type | No. parameters | Transform cost |
|---|---|---|
| Model-based | $PMn + Mn + M$ | $PMn$ |
| Transform-based | $PQ(n^2+n) + 2Mn + M$ | $PQ(n^2+n) + M(n^2+n)$ |

Table 1: Comparison of the number of parameters and adaptation cost in model-based and transform-based CAT

Table 1 shows the number of parameters for each of the cluster representations (including the shared variances and priors). For the typical system examined in this paper, $M = 32626$, $P = 8$, $n = 39$, $Q = 32$, there are 8,939,524 parameters in the model-based cluster and 1,671,774 in the transform-based system. Thus a reduction by a factor of 5 in the number of parameters is possible by using transform-based clusters rather than model-based clusters. By reducing the number of parameters per cluster, the number of clusters that may be robustly estimated is increased. Also for a given number of clusters the storage requirements for the system are reduced. The negative side of this reduction in the number of parameters is the accuracy in the representation of the clusters. Transform-based clusters rely on a simple piecewise linear transformation to model a particular cluster. Furthermore, there is some additional cost in generating the mean values for a particular speaker. Using model-based clusters about 10 million multiply accumulates are required to adapt the typical model set described above. Whereas, over 50 million multiply accumulates are used for the transform-based clusters.

## 5.2 Training Initialisation

One important issue in any training scheme is how to set the initial model parameters. For both forms of CAT there are two distinct initialisation schemes. The first uses hypothesised cluster weight vectors from the training data. The second uses cluster means from the training data. In both schemes the posterior probability of a particular Gaussian component at a particular time instance, $\gamma_m(\tau)$, is also required. This is normally obtained from an SI model set.

### 5.2.1 Cluster weight vector initialisation

An initial cluster weight vector for all the speakers in the training data is required. One approach to this problem is to use a bottom up clustering scheme based on simple speaker-dependent models derived from the training data. The complexity of these speaker-dependent models may vary from a single Gaussian to context dependent triphones depending on the amount of training data per speaker. Having generated a model for each speaker in the training data, agglomerative clustering is then be used to group the speakers into the required number of clusters. The output from this clustering process is a unique assignment of a speaker to a cluster. This is then converted into a cluster weight vector, $\boldsymbol{\lambda}^{(sr)}$, with a single non-zero entry (of value 1). Using $\gamma_m(\tau)$ estimated from the SI model set, the sufficient statistics $\boldsymbol{G}^{(m)}$, $\boldsymbol{K}^{(m)}$ and $\boldsymbol{L}^{(m)}$, may be calculated as defined by equations 16, 17 and 18.

Recently an alternative scheme for very rapid adaptation has been proposed, *eigenvoices* [9]. This scheme may also be used to initialise CAT. Again a set of simple speaker-dependent models are generated for every speaker in the training data. The means of each speaker are concatenated into a single meta-vector. Eigenvalue decomposition is then performed on the covariance matrix derived from all the training speaker meta-vectors to yield a set of eigenvalues and eigenvectors. The subset of eigenvectors associated with the largest eigenvalues is selected as the initial cluster means (along with the global mean vector). It should be noted that these cluster means are only for the simplified models, not the final models required. The problem is now to convert the set of simplified model cluster means to a form that can be used for the more complex final models. There are two choices, a projection scheme, or an ML approach can be used. Both schemes are described in [9] for using eigenvoices.

For projection based schemes the meta-vector of speaker $s$, $\boldsymbol{v}^{(s)}$, may be written as

$$\boldsymbol{v}^{(s)} = \overline{\boldsymbol{v}} + \sum_{p} \boldsymbol{v}'^{(s)T} \boldsymbol{e}^{(p)} \boldsymbol{e}^{(p)} + \boldsymbol{\epsilon}^{(s)} \tag{34}$$

where $\overline{\boldsymbol{v}}$ is the mean of the meta-vectors, $\boldsymbol{v}'^{(s)} = \boldsymbol{v}^{(s)} - \overline{\boldsymbol{v}}$, $\boldsymbol{e}^{(p)}$ is the $p^{th}$ eigenvector (ordered according to the eigenvalues) of $\sum_s (\boldsymbol{v}^{(s)} - \overline{\boldsymbol{v}})(\boldsymbol{v}^{(s)} - \overline{\boldsymbol{v}})^T$ and $\boldsymbol{\epsilon}^{(s)}$ is the residual error for speaker $s$[12]. Thus the cluster weight vector becomes (assuming there is only one cluster weight class)

$$\boldsymbol{\lambda}^{(s1)} = \left[ \begin{array}{cccc} \boldsymbol{v}'^{(s)T} \boldsymbol{e}^{(1)} & \dots & \boldsymbol{v}'^{(s)T} \boldsymbol{e}^{(P-1)} & 1 \end{array} \right]^T \tag{35}$$

In the ML approach a simple SI model (of the same complexity as the SD models) is used to obtain the posterior probability of a particular Gaussian component for each frame, $\gamma_m(\tau)$. This allows $\boldsymbol{G}_{\boldsymbol{w}}^{(s1)}$ and $\boldsymbol{k}_{\boldsymbol{w}}^{(s1)}$ to be found for each training speaker using $\boldsymbol{M}^{(m)}$ defined as

$$\boldsymbol{M}^{(m)} = \left[ \begin{array}{cccc} \boldsymbol{e}_m^{(1)} & \dots & \boldsymbol{e}_m^{(P-1)} & \overline{\boldsymbol{v}}_m \end{array} \right] \tag{36}$$

where $\boldsymbol{e}_m^{(p)}$ are the elements of meta-vector $\boldsymbol{e}^{(p)}$ associated with Gaussian component $m$. The cluster vector $\boldsymbol{\lambda}^{(s1)}$ may then be calculated using equation 11. It is interesting to note that the use of eigenvoices naturally results in a bias cluster given by the meta-vector mean, in contrast to the simple speaker clustering scheme where the bias vector would have to be added. Using the speaker cluster weight vectors and an SI model set[13] a new canonical model may then be estimated.

#### 5.2.2 Cluster mean initialisation

In some situations it is simpler to initialise the cluster means rather than the cluster weight vector, in particular if some knowledge about the variability of the data is known *a-priori*. An example of this is the ARPA Hub4 training data. Approximately half of the training data has detailed labels of speaker and *focus* condition. The other half simply has speaker marks. Hence if speaker/focus models are required, for example male or female, noisy or quiet environment, it is possible to initially partition only half the training data. A simple scheme for generating the cluster means from the fully labelled subset of training data is to adapt the SI models to the required subset. For example, MLLR or maximum *a-posteriori* [8] adaptation may be used. Using these means and the SI model for the posterior probabilities (and initial Gaussian component variance values), which give an initial canonical model, a cluster weight vector for each speaker may be estimated. The CAT training scheme may then be run.

### 5.3 Recognition with CAT

During recognition, for each test speaker, s, the set of $R$ cluster weight vectors must be estimated. In a similar fashion to the estimation of the cluster weight vectors for the training speakers this is performed in an EM fashion where the auxiliary function $\mathcal{Q}(\boldsymbol{\Lambda}; \mathcal{M}, \hat{\boldsymbol{\Lambda}})$ defined in equation 8 is optimised with respect to $\boldsymbol{\Lambda}$. Here there is only a single test speaker, thus

$$\boldsymbol{\Lambda} = \left\{ \{\boldsymbol{\lambda}^{(s1)}, \dots, \boldsymbol{\lambda}^{(sR)}\} \right\} \tag{37}$$

[12] Thus the set of eigenvectors have been selected to minimise $\sum_s \boldsymbol{\epsilon}^{(s)T} \boldsymbol{\epsilon}^{(s)}$.

[13] This SI model set may either be based on the simple models or the full models as they are only used to obtain the alignments.

The estimation of $\mathbf{\Lambda}$ simply requires using equation 11. Since this is an EM scheme, a technique for initialising the complete dataset must be devised. The most commonly used scheme in adaptive training is to use an SI model set to calculate the posterior probabilities, $\gamma_m(\tau)$, for the complete dataset. In the case of unsupervised adaptation the SI model set is also used to recognise the adaptation data to obtain the hypothesised transcription. Once the posterior probabilities have been initialised, $\mathbf{\Lambda}$ may be computed. Having obtained an initial estimate of $\mathbf{\Lambda}$, the estimation scheme may be repeated using the estimate of $\mathbf{\Lambda}$ to obtain the new posterior distributions, $\gamma_m(\tau)$, and transcription hypothesis if required.

It is interesting at this point to compare the number of parameters required to be estimated for CAT to those of other rapid adaptation schemes. For very little adaptation data, a single CAT cluster weight vector may be estimated for the test speaker. This requires estimating $P$ cluster weight vector parameters (or $P-1$ parameters when a bias cluster is used). For the linear transform case, the simplest linear transform that may be applied to the models is a single bias vector added to the mean. For an $n$-dimensional feature vector, this requires estimating $n$ bias parameters. Since the number of clusters, $P$, is typically far smaller than the dimensionality of the observation vector, $n$, CAT requires fewer parameters than even the simplest linear transform adaptation scheme. As the complexity of the inear transform increases the number of parameters in the transforms increases rapidly. For example for a full MLLR transform there are $n^2+n$ transform parameters to estimate. Comparing CAT with MAP-based schemes, CAT allows all the model means to be adapted with very few parameters. Standard MAP only adapts the observed Gaussian components. In the case of large vocabulary systems with one adaptation sentence, only a small fraction of Gaussian components will thus be adapted. Extensions to MAP are possible, such as structured MAP (SMAP) [18]. SMAP allows unobserved Gaussian components to be adapted, but requires linear transforms to be estimated. As previously mentioned, even the simplest linear transforms typically require more parameters than CAT.

### 5.4 Combination with other Adaptation Schemes

The primary use of CAT is as a rapid adaptation scheme for scenarios where there is very little adaptation data available. Alternatively CAT may be used to provide the canonical model for other adaptation schemes such as MLLR. In contrast to MLLR where incorporating the transform into an adaptive training scheme may become prohibitively expensive, CAT allows a simple transform to be efficiently used in the adaptive training scheme. To use CAT for the canonical model for other more powerful adaptation schemes, first the cluster weight vectors for the test speaker are estimated. The canonical model is then adapted. The more complicated transform are applied on top of this CAT-adapted model. Thus in the case of global MLLR adaptation the transform for speaker $s$, $\boldsymbol{W}^{(s1)}$, is determined using the CAT adapted speaker means, $\boldsymbol{M}^{(m)}\boldsymbol{\lambda}^{(sr_m)}$. The final adapted mean of Gaussian component $m$ for speaker $s$, $\boldsymbol{\mu}^{(sm)}$, will be

$$\boldsymbol{\mu}^{(sm)} = \boldsymbol{A}^{(s1)}\boldsymbol{M}^{(m)}\boldsymbol{\lambda}^{(sr_m)} + \boldsymbol{b}^{(s1)} \tag{38}$$

As the amount of adaptation data increases, the possible gains from using the CAT canonical model decreases[14]. For example as the amount of adaptation data becomes very large the final speaker dependent models using MAP are independent of the original canonical model adapted.

## 6 Bias Clusters for Linear Transforms

In this paper a bias cluster to represent any speaker independent aspect of the mean vector was introduced. This concept of a bias cluster is also relevant to other linear transform adaptation schemes. Consider SAT [1]. Here the adapted mean for Gaussian component $m$ of speaker $s$ is given by

$$\boldsymbol{\mu}^{(sm)} = \boldsymbol{A}^{(st_m)}\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} + \boldsymbol{b}^{(st_m)} \tag{39}$$

[14]This is true of any adaptive training scheme.

where $\boldsymbol{A}^{(st_m)}$ and $\boldsymbol{b}^{(st_m)}$ are the speaker dependent transform and $\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)}$ is the canonical model mean of Gaussian component $m$. The regression of the model means is based about some vector $\boldsymbol{b}^{(sq)}$ for *all* Gaussian components in transform class $q$. In some circumstances this could be a restriction on the ability of the system to adapt.

Using bias clusters a more general formulation of MLLR[15] may be used. The adapted mean of Gaussian component $m$ for speaker $s$ becomes

$$\boldsymbol{\mu}^{(sm)} = \boldsymbol{A}^{(st_m)}\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} + \boldsymbol{b}^{(st_m)} + \boldsymbol{\mu}_{\boldsymbol{b}}^{(m)} \tag{40}$$

where the bias cluster mean for Gaussian component $m$, $\boldsymbol{\mu}_{\boldsymbol{b}}^{(m)}$, is that component of the mean vector which is independent of the speaker. This bias cluster may either be model-based, or transform-based where

$$\boldsymbol{\mu}_{\boldsymbol{b}}^{(m)} = \boldsymbol{A}^{(bt_m^{(b)})}\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} + \boldsymbol{b}^{(bt_m^{(b)})} \tag{41}$$

where $t_m^{(b)}$ is the bias transform class of Gaussian component $m$. In effect this allows the point in acoustic space about which the regression is based to be specific to a particular Gaussian component. $\boldsymbol{b}^{(st_m)}$ is kept in order to give flexibility to the transform, for example when modelling convolution noise in the cepstral domain [17]. During recognition the standard MLLR parameters are estimated and applied to the speaker-dependent component of the mean and then the bias cluster added.

The training of this form of model is a simple modification to the standard SAT training of HMMs [1]. An iterative procedure is used.

1. The transform for a particular speaker is computed given the current estimate of the mean and bias cluster. This transform estimation is identical to standard MLLR, except that instead of using the observation $\boldsymbol{o}(\tau)$ the transformed observation $\boldsymbol{o}(\tau) - \boldsymbol{\mu}_{\boldsymbol{b}}^{(m)}$ is used for Gaussian component $m$.

2. The mean is estimated using the standard SAT formulae, except again the transformed observation is used rather than the standard observation. The mean is thus a function of the transform and the bias cluster mean.

3. The bias cluster is then re-estimated. Given the canonical model and the current estimates of the speaker transform, the estimate of the bias cluster is simple. Using standard mean estimation formulae for model-based biases

$$\boldsymbol{\mu}_{\boldsymbol{b}}^{(m)} = \frac{\sum_{s,\tau}\gamma_m(\tau)\left(\boldsymbol{o}(\tau) - \left(\boldsymbol{A}^{(st_m)}\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} + \boldsymbol{b}^{(st_m)}\right)\right)}{\sum_{s,\tau}\gamma_m(\tau)} \tag{42}$$

   If transform-based biases are used then the standard MLLR estimation scheme may be used.

4. Go to (2) until the convergence criterion satisfied.

5. Go to (1) until the convergence criterion satisfied.

The simplest initialisation scheme for $\boldsymbol{\mu}_{\boldsymbol{b}}^{(m)}$ is to initialise it to zero. This form of model will be examined in future work.

[15]The same scheme may be applied to the constrained model-space transform, or feature space transform [7]. The transformed observation for Gaussian component $m$ belonging to transform class $q$ for speaker $s$ would be given by $\hat{\boldsymbol{o}}^{(sm)}(\tau) = \boldsymbol{A}^{(st_m)}\boldsymbol{o}(\tau) + \boldsymbol{b}^{(st_m)} - \boldsymbol{\mu}_{\boldsymbol{b}}^{(m)}$. The re-estimation routine is similar to that for MLLR.

# 7 Results

The task used to evaluate the CAT system was an internal IBM dictation task. It is a speaker-independent task using read speech data recorded in clean environments with the same microphone. There are 1670 speakers in the training data with a total of 36272 training sentences. A state clustered decision tree system was used throughout with 2755 states. The front end consisted of 12 cepstra, plus the zeroth cepstra with delta and delta-delta parameters appended. Cepstral mean normalisation was applied to the feature vector. The test set consists of 9 speakers, each uttering 61 sentences giving a total of around 10,000 words in the test set. The results quoted are the average word error rates over all test speakers. A trigram language model was used in all tests. Two basic model sizes were considered. The first used four Gaussian components per state and the second an average of 12 Gaussian components per state. The cluster vectors were estimated in a supervised adaptation mode using 20 adaptation sentences unless otherwise stated. The adaptation sentences were selected from held out data from each test speaker. All experiments used a single cluster weight class unless otherwise stated.

The training procedure was initialised using the cluster weight vector initialisation described in section 5.2.1. Simple monophone models were built for each speaker. These were then agglomeratively clustered using a simple likelihood distance measure until the required number of clusters were obtained. This assignment of speaker to cluster was used to initialise the cluster weight vector for each speaker. The SI model set was used to initialise the posterior probabilities. Four iterations of updating the canonical model and the cluster weight vector for each speaker were performed. For the first set of experiments a simple 2 cluster system, one cluster being a bias clusters was built. In this case the speakers were initialially clustered into two groups. These two groups were then initialised with cluster weight vectors $\begin{bmatrix} -1 & 1 \end{bmatrix}^T$ and $\begin{bmatrix} 1 & 1 \end{bmatrix}^T$.

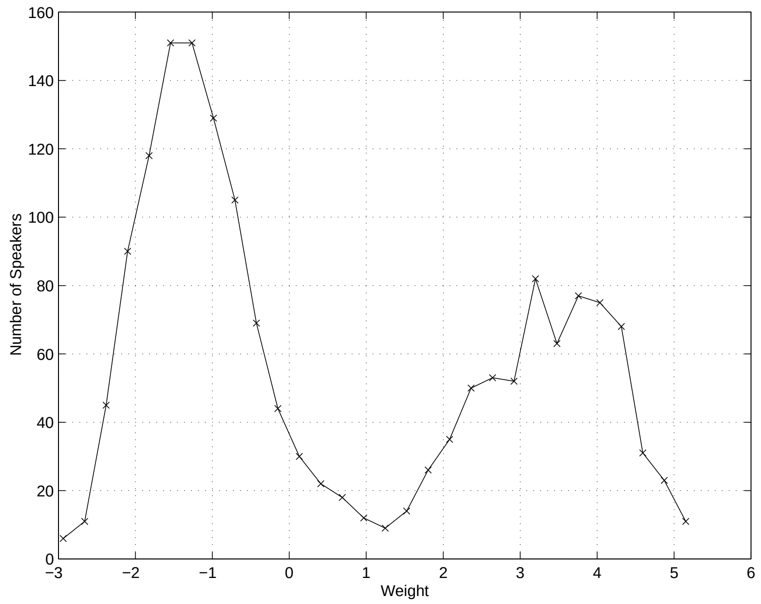


Figure 3: Distribution of cluster weight vector values on the training speakers

Figure 3 shows the distribution of the interpolation weights for all the training speakers in the simple 2 cluster system. Since the second cluster is a bias cluster, there is only a single free parameter, which is plotted in figure 3. The distribution of the values is distinctly bimodal, which roughly correspond to “male” and “female” modes. However in contrast to standard gender dependent models, which effectively assume a delta function at two values in the system, the two modes have large variances[16].

| No. Gaussian Components | No. of Clusters | Word Error Rate |
|---|---|---|
| 4 | SI | 15.1 |
| | 1 (+1) | 12.9 |
| | 8 | 12.3 |
| 12 | SI | 12.6 |
| | 1 (+1) | 11.7 |
| | 8 | 11.5 |

Table 2: Comparison of the baseline systems with the CAT systems. *1 (+1)* indicates a two cluster system where one of the clusters is a bias cluster. *SI* indicates the speaker independent performance.

Table 2 shows the performance of the baseline CAT systems. Two forms of model-based cluster CAT were examined. The first used the simple 2 cluster system, labelled *1 (+1)*, thus there was only a single free parameter. The second used an 8 cluster system with all cluster vectors depending on the speaker, *8*. As expected the SI 4 Gaussian component system performed significantly worse than the SI 12 Gaussian component system. In both cases the use of CAT reduced the word error rate. The gains for the smaller system were larger than those of the 12 Gaussian component system. For example, on the 8 cluster system CAT reduced the word error rate by 19% relative on the 4 Gaussian component system and only 9% on the 12 Gaussian component system. It is interesting to note that a simple 8 cluster 4 Gaussian component system achieves the same performance as the SI 12 Gaussian component system, despite using a third of the number of Gaussian components during recognition.

As an alternative to the soft use of clusters the standard hard clustering scheme may be used. A 4 Gaussian component system with hard cluster selection was built with two clusters. The Gaussian component priors and variances were tied in the same fashion as in CAT. This system is very similar to tied version of the standard gender-dependent systems. However, here the "gender" is determined in an ML fashion. This system has a single binary parameter to be estimated for each test speaker. The word error rate for this hard clustering system was 13.4%. This may be compared with the *1(+1)* soft clustering scheme which also uses two clusters and has a single variable to be estimated for each test speaker. The *1(+1)* system had a word error rate of 12.9%. This indicates that the use of interpolation weights rather than hard assignments gives some gain in performance for the same number of clusters.

| No. Gaussian Components | Number Clusters | Type of Clusters | Word Error Rate |
|---|---|---|---|
| 4 | 1(+1) | model-based | 12.9 |
| | | transform-based | 13.3 |
| 12 | 1(+1) | model-based | 11.7 |
| | | transform-based | 11.8 |
| | 8 | model-based | 11.5 |
| | | transform-based | 11.3 |

Table 3: Comparison of transform-based and model-based clusters. *1 (+1)* indicates a two cluster system where one of the clusters is a bias cluster.

[16]By arbitrarily scaling all the speaker-dependent cluster means the variances may be made arbitrarily large. For the system built the means of the bias cluster and the speaker-dependent cluster were found to have about the same range.

Table 3 compares the performance of model-based and transform-based clusters. For each transform-based cluster 32 full MLLR transforms, with acoustic-space clustered transform classes, were used. For small numbers of clusters the model-based schemes outperform the transform-based schemes, since it is possible to robustly estimate the cluster means. However, as discussed in section 4, transform-based clusters allow a far more compact model to be built. This reduction in the number of parameters becomes important as the number of clusters and Gaussian components increases. Using the 12 Gaussian component system with 8 clusters, the transform-based clusters slightly out-performed the model-based clusters. This illustrates the ability to use more clusters with the transform-based scheme, since there are fewer free parameters to estimate from the training data.

| No. Gaussian Components | Adaptation Scheme | No Adapt. Sent. | | | |
|---|---|---|---|---|---|
| | | 1 | 5 | 10 | 20 |
| 12 | CAT (8 trans) | 11.6 | 11.2 | 11.0 | 11.3 |
| | CAT (8 trans mult) | 11.6 | 11.0 | 11.0 | 10.8 |
| 12 | MLLR (full) | 13.6 | 11.5 | 11.0 | 10.5 |
| | MLLR (block) | 12.2 | 11.5 | 11.1 | 10.8 |

Table 4: CAT and MLLR recognition performance with number of adaptation sentences compared with the speaker-independent performance of 12.6%. *8 trans* indicates the use of 8 transform-based clusters with a single weight classes, *8 trans mult* the use of multiple weight classes.

One of the advantages of CAT is that few parameters are used to adapt the models. Table 4 compares the performance of the 8 transform-based 12 Gaussian component system to using MLLR with block diagonal or full transforms with varying numbers of adaptation sentences. For the block-diagonal transform, three blocks were selected, the static, delta and delta-delta parameters. As expected the performance of the full MLLR transform with only a single adaptation sentence, typically around 600 frames of data, was poor. In fact the use of the transform degraded the performance compared to the SI baseline of 12.6%. Using a block-diagonal transform gave slight improvements[17]. CAT gave the best performance using a single sentence. All systems showed improvements as the amount of adaptation data increased. Other than for the single sentence case, where a single transform was forced to be generated, the number of MLLR transforms was determined using a regression class tree with minimum occupancy counts determined empirically on a similar task. What is interesting to note is that a single CAT cluster weight vector performed about the same as MLLR up to around 10 sentences. However it should be emphasised that the MLLR transforms used an SI model set as the canonical model, whereas the CAT system was adaptively trained[18]. Additional slight gains were observed using multiple cluster weight classes (*8 trans mult*). In this case, the same regression class was used for the cluster weight classes as for the transform classes.

CAT models may also be used as the canonical models in conjunction with other transforms. The 12 Gaussian component model-based CAT *1(+1)* system was used for adaptation with a single global constrained model-space transform [7] and multiple MLLR transforms estimated on 50 adaptation sentences. This yielded a word error rate of 9.4%. This can be compared with using an SI model as the canonical model with the same linear transformation, though of course no cluster smoothing, which gave a word error rate of 10.0%. This performance gain is roughly consistent with other adaptive training schemes.

[17] The block-diagonal MLLR transform gave slightly better performance than a diagonal transform.

[18] Though there is some inconsistency in the canonical model used (one adaptively trained the other not), this is due to the practicalities of building a standard SAT system on training data from 1670 speakers.

## 8 Conclusions

This paper has introduced a new form of adaptive training, cluster adaptive training. The scheme uses a very simple representation of a speaker, a cluster weight vector, to smooth a set of cluster means. It may be viewed as an extension of speaker clustering, or as a restricted version of speaker adaptive training. Two forms of cluster representation are described. The first uses sets of means themselves. The second, a far more compact representation, uses a canonical mean and MLLR transforms of that canonical mean to represent a cluster. Re-estimation formulae are given for both the cluster weight vectors and the cluster parameters. Both forms of cluster are found to reduce the word error rate. The use of CAT in combination with other linear adaptation schemes shows gains over adapting speaker independent models.

Future work will look at the use of bias clusters with SAT. In addition alternative transforms for the transform-based CAT scheme will be examined. Efficient schemes to efficiently interpolate cluster variances also need to be derived.

# A Cluster Parameter Estimation

The estimation of the cluster weight vector and the MLLR transforms in the transform-based clusters are derived in [4, 5]. This appendix details the mean cluster estimation for model-based clusters and the canonical model for transform-based clusters.

## A.1 Model-Based Clusters

The set of mean parameters are estimated in an EM fashion. The auxiliary function to be maximised with respect to cluster means (ignoring the standard constants) is given by

$$\mathcal{Q}(\mathcal{M};\hat{\mathcal{M}},\mathbf{\Lambda}) = -\frac{1}{2}\sum_{s,m,\tau}\gamma_m(\tau)\left(\boldsymbol{o}(\tau)-\boldsymbol{M}^{(m)}\boldsymbol{\lambda}^{(sr_m)}\right)^T\mathbf{\Sigma}^{(m)-1}\left(\boldsymbol{o}(\tau)-\boldsymbol{M}^{(m)}\boldsymbol{\lambda}^{(sr_m)}\right) \tag{43}$$

Differentiating this expression with respect to $\boldsymbol{M}^{(m)}$ and equating to zero yields

$$\mathrm{vec}(\boldsymbol{Z}^{(m)}) = \left(\mathbf{\Sigma}^{(m)-1}\otimes\boldsymbol{G}^{(m)}\right)\mathrm{vec}(\boldsymbol{M}^{(m)}) \tag{44}$$

where

$$\boldsymbol{Z}^{(m)} = \mathbf{\Sigma}^{(m)-1}\sum_{s,\tau}\gamma_m(\tau)\boldsymbol{o}(\tau)\boldsymbol{\lambda}^{(sr_m)T} = \mathbf{\Sigma}^{(m)-1}\boldsymbol{K}^{(m)T} \tag{45}$$

$\boldsymbol{G}^{(m)}$ and $\boldsymbol{K}^{(m)}$ are defined in equations 16 and 17 respectively, and vec() performs a row-wise mapping of a matrix into a vector[19]. Thus the set of means may be simply calculated. This expression is greatly simplified for the diagonal covariance matrix case. Here a set of independent equations result where for row $i$ of the mean matrix $\boldsymbol{M}^{(m)}$, $\boldsymbol{m}_i^{(m)}$, (a $1\times P$ vector),

$$\frac{1}{\sigma_{ii}^{(m)2}}\left(\boldsymbol{k}^{(m)T}\right)_i = \boldsymbol{m}_i^{(m)}\frac{1}{\sigma_{ii}^{(m)2}}\boldsymbol{G}^{(m)} \tag{47}$$

where $\left(\boldsymbol{k}^{(m)T}\right)_i$ is the $i^{th}$ row of the transpose of $\boldsymbol{K}^{(m)}$, again a $1\times P$ vector. Hence

$$\boldsymbol{m}_i^{(m)} = \left(\boldsymbol{k}^{(m)T}\right)_i\boldsymbol{G}^{(m)-1} \tag{48}$$

Rearranging and noting that $\boldsymbol{G}^{(m)}$ is symmetric gives

$$\boldsymbol{M}^{(m)T} = \boldsymbol{G}^{(m)-1}\boldsymbol{K}^{(m)} \tag{49}$$

The variance is found using

$$\mathbf{\Sigma}^{(m)} = \frac{\sum_{s,\tau}\gamma_m(\tau)(\boldsymbol{o}(\tau)-\boldsymbol{\mu}^{(sm)})(\boldsymbol{o}(\tau)-\boldsymbol{\mu}^{(sm)})^T}{\sum_{s,\tau}\gamma_m(\tau)} \tag{50}$$

where $\boldsymbol{\mu}^{(sm)} = \boldsymbol{A}^{(sm)}\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} + \boldsymbol{b}^{(sm)}$. This may be easily rearranged to give

$$\mathbf{\Sigma}^{(m)} = \frac{\boldsymbol{L}^{(m)} - \boldsymbol{M}^{(m)}\boldsymbol{K}^{(m)}}{\sum_{s,\tau}\gamma_m(\tau)} \tag{51}$$

[19] As an example for the case described here

$$\mathrm{vec}(\boldsymbol{Z}^{(m)}) = \begin{bmatrix}\boldsymbol{z}_1^{(m)} & \ldots & \boldsymbol{z}_n^{(m)}\end{bmatrix}^T \tag{46}$$

## A.2 Transform-Based Clusters

The estimation of the transform-based cluster parameters as previously described is done iteratively. First a set of cluster dependent transforms is estimated, $\mathcal{T}$, given the current model parameters. Second the model parameters are updated. The transform estimation is identical to the estimation of the transforms in transformation smoothing [4, 5]. This section will just describe the estimation of the model parameters $\tilde{\mathcal{M}}$. From equation 15 the following auxiliary function must be optimised with respect to $\tilde{\mathcal{M}}$

$$\mathcal{Q}(\tilde{\mathcal{M}};\mathcal{T},\hat{\mathcal{M}},\mathbf{\Lambda}) = -\frac{1}{2}\sum_{s,m,\tau}\gamma_m(\tau)\left(\boldsymbol{o}(\tau)-\boldsymbol{W}^{(sm)}\boldsymbol{\xi}_{\boldsymbol{a}}^{(m)}\right)^T\mathbf{\Sigma}^{(m)-1}\left(\boldsymbol{o}(\tau)-\boldsymbol{W}^{(sm)}\boldsymbol{\xi}_{\boldsymbol{a}}^{(m)}\right) \quad (52)$$

where $\boldsymbol{W}^{(sm)} = \sum_p \lambda_p^{(sr_m)}\boldsymbol{W}_{\boldsymbol{c}}^{(pt_m)}$, $\gamma_m(\tau)$ is the posterior probability of Gaussian component $m$ at time $\tau$ determined using the old model parameters $\hat{\mathcal{M}}$ the cluster vectors $\mathbf{\Lambda}$ and the transform set, $\mathcal{T}$. Comparing this expression with the standard SAT expressions in [1], noting that the transform for a particular Gaussian component of a speaker in the training data is $\boldsymbol{W}^{(sm)}$, ($\boldsymbol{W}^{(sm)} = \begin{bmatrix}\boldsymbol{A}^{(sm)} & \boldsymbol{b}^{(sm)}\end{bmatrix}$), then the canonical mean, $\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)}$, is given by

$$\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} = \left(\sum_{s,\tau}\gamma_m(\tau)\boldsymbol{A}^{(sm)T}\mathbf{\Sigma}^{(m)-1}\boldsymbol{A}^{(sm)}\right)^{-1}\sum_{s,\tau}\gamma_m(\tau)\boldsymbol{A}^{(sm)T}\mathbf{\Sigma}^{(m)-1}\left(\boldsymbol{o}(\tau)-\boldsymbol{b}^{(sm)}\right) \quad (53)$$

and variance, $\mathbf{\Sigma}^{(m)}$, is again given by equation 50. For CAT $\boldsymbol{A}^{(sm)} = \sum_p \lambda_p^{(sr_m)}\boldsymbol{A}_{\boldsymbol{c}}^{(pt_m)}$ and $\boldsymbol{b}^{(sm)} = \sum_p \lambda_p^{(sr_m)}\boldsymbol{b}_{\boldsymbol{c}}^{(pt_m)}$, so rewriting equation 53 and using equations 16 and 17 gives

$$\boldsymbol{\mu}_{\boldsymbol{a}}^{(m)} = \left(\sum_{i,j} g_{ij}^{(m)}\boldsymbol{A}_{\boldsymbol{c}}^{(it_m)T}\mathbf{\Sigma}^{(m)-1}\boldsymbol{A}_{\boldsymbol{c}}^{(jt_m)}\right)^{-1}\sum_{p}\boldsymbol{A}_{\boldsymbol{c}}^{(pt_m)T}\mathbf{\Sigma}^{(m)-1}\left(\boldsymbol{k}_p^{(m)T}-\sum_q g_{pq}^{(m)}\boldsymbol{b}_{\boldsymbol{c}}^{(pt_m)}\right) \quad (54)$$

Assuming a diagonal covariance matrix the expression for the variance may be rearranged as

$$\mathbf{\Sigma}^{(m)} = \mathrm{diag}\left(\frac{\boldsymbol{L}^{(m)} - 2\sum_p \boldsymbol{W}_{\boldsymbol{c}}^{(pt_m)}\boldsymbol{\xi}_{\boldsymbol{a}}^{(m)}\boldsymbol{k}_p^{(m)} + \sum_{i,j} g_{ij}^{(m)}\boldsymbol{W}_{\boldsymbol{c}}^{(it_m)}\boldsymbol{\xi}_{\boldsymbol{a}}^{(m)}\boldsymbol{\xi}_{\boldsymbol{a}}^{(m)T}\boldsymbol{W}_{\boldsymbol{c}}^{(jt_m)T}}{\sum_{s,\tau}\gamma_m(\tau)}\right) \quad (55)$$

The estimation of the canonical model for the transform-based clusters is thus iterative since the variance estimate is a function of the mean and the mean estimate is dependent on the variance.

## References


[1] T Anastasakos, J McDonough, R Schwartz, and J Makhoul. A compact model for speaker-adaptive training. In *Proceedings ICSLP*, pages 1137–1140, 1996.

[2] A P Dempster, N M Laird, and D B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

[3] V V Digalakis, D Rtischev, and L G Neumeyer. Speaker adaptation using constrained estimation of Gaussian mixtures. *IEEE Transactions Speech and Audio Processing*, 3:357–366, 1995.

[4] M J F Gales. The generation and use of regression class trees for MLLR adaptation. Technical Report CUED/F-INFENG/TR263, Cambridge University, 1996. Available via anonymous ftp from: svr-ftp.eng.cam.ac.uk.

[5] M J F Gales. Transformation smoothing for speaker and environmental adaptation. In *Proceedings Eurospeech*, pages 2067–2070, 1997.

[6] M J F Gales. Cluster adaptive training for speech recognition. In *Proceedings ICSLP*, pages 1783–1786, 1998.

[7] M J F Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12:75–98, 1998.

[8] J L Gauvain and C H Lee. Maximum a-posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions Speech and Audio Processing*, 2:291–298, 1994.

[9] R Kuhn, P Nguyen, J-C Junqua, L Goldwasser, N Niedzielski, S Fincke, K Field, and M Contolini. Eigenvoices for speaker adaptation. In *Proceedings ICSLP*, pages 1771–1774, 1998.

[10] L Lee and R C Rose. Speaker normalisation using efficient frequency warping procedures. In *Proceedings ICASSP*, volume 1, pages 353–356, 1996.

[11] C J Leggetter. *Improved Acoustic Modelling for HMMs using Linear Transformations.* PhD thesis, Cambridge University, 1995.

[12] C J Leggetter and P C Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density HMMs. *Computer Speech and Language*, 9:171–186, 1995.

[13] S Matsoukas, R Schwartz, H Jin, and L Nguyen. Practical implementations of speaker-adaptive training. In *Proceedings DARPA Speech Recognition Workshop*, 1997.

[14] M Padmanabhan, L R Bahl, D Nahamoo, and M A Picheny. Speaker clustering and transformation for speaker adaptation in speech recognition systems. *IEEE Transactions Speech and Audio Proessing*, pages 71–77, 1998.

[15] D Pye and P C Woodland. Experiments in speaker normalisation and adaptation for large vocabulary speech recognition. In *Proceedings ICASSP*, pages 1047–1050, 1997.

[16] A Sankar, F Beaufays, and V Digalakis. Training data clustering for improved speech recognition. In *Proceedings Eurospeech*, pages 502–505, 1995.

[17] A Sankar and C H Lee. A maximum likelihood approach to stochastic matching for robust speech recognition. *IEEE Transactions Speech and Audio Processing*, 4:190–202, 1996.

[18] K Shinoda and C-H Lee. Structural MAP speaker adaptation using hierarchical priors. In *Proceedings IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 381–388, 1997.