CAMBRIDGE UNIVERSITY
ENGINEERING DEPARTMENT

TRANSFORMATION STREAMS AND THE HMM ERROR MODEL

M.J.F. Gales

CUED/F-INFENG/TR.416

July 2001

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

Abstract

The most popular model used in automatic speech recognition is the hidden Markov model (HMM). Though good performance has been obtained with such models there are well known limitations for its ability to model speech. A variety of modifications to the standard HMM topology have been proposed to handle these problems. One such scheme is the factorial HMM. This paper introduces a new form of factorial HMM which makes use of *transformation streams*. This new scheme is a generalisation of the standard factorial HMM and other related schemes in speech processing. A particular form of this model, the *HMM error model* (HEM) is described in detail. The HEM is evaluated on two standard large vocabulary speaker independent speech recognition tasks. On both tasks significant reductions in word error rate are obtained over standard HMM-based systems.

1 Introduction

One of the major problems in automatic speech recognition is to generate an acoustic model that performs well on unseen data[1] and is compact. The system should be compact to allow the recognition to be performed in "reasonable" time and to enable robust estimation of the model parameters. Over the years this has lead to a variety of modifications to the standard hidden Markov model (HMM) to overcome the limitations of the model for speech recognition. One form that is currently popular is to view the HMM as a dynamic version of a Bayesian network [24]. In particular schemes based on factorial HMMs [13][2] have been examined [17]. Techniques that have been investigated include convolutional HMMs [18], multi-band systems [19], dynamic Bayesian networks [32], multiple stream systems [30] and loosely coupled HMMs [22]. This paper introduces a new form of factorial HMM which makes use of *transformation streams*. Transformation streams allow the state of one stream to modify the model parameters, or features space, of other streams. This concept of model and feature transformation is very common (and successful) in speech recognition for speaker adaptation [16, 8] and covariance modelling [9]. This paper incorporates these transformation into a multiple stream framework. The use of transformations is shown to be a generalisation of standard factorial HMMs. This paper examines the form of the multiple stream models rather than details of inference of factorial models. For further details of inference with factorial models, and fast approximations, see [13].

In addition to presenting a new form of stream model, a novel acoustic model for speech recognition is described, the *HMM error model* (HEM). The model makes use of a transformation stream in conjunction with a simple, single state, "model" stream. Due to the nature of the model it may also be interpreted as a dynamic filter, the transformation stream, and a residual model. This allows a simple interpretation of how the model can improve the modelling ability of a standard HMM. Rather than relying on the residual being accurately modelled by a zero mean, identity covariance matrix, Gaussian distribution, the standard HMM assumption, it may be explicitly modelled using any standard probabilistic model. In particular the use of Gaussian mixture models is investigated. In common with many other acoustic modelling schemes and factorial HMM schemes, the HEM may be described as a form of *soft* parameter tying. In soft tying schemes model parameters are "related" to one another. In contrast standard tying schemes [30] require that model parameters are either independent or identical. Examples include speaker adaptive training [1], soft state tying [15] and semi-tied covariance matrices [9].

This paper is organised as follows. The next section will describe factorial HMMs and various forms of stream representations that have previously been investigated. Transformation streams are then introduced and described. In addition an extension to multiple transformation streams is then detailed. The HEM is described along with and an interpretation of the model as a dynamic data filter and residual model. Finally experiments on two large vocabulary speech recognition tasks are used to illustrate the advantage of HEMs over standard HMMs.

2 Factorial Modelling Schemes

This section will briefly review factorial modelling schemes[3]. In recent years these schemes have become very popular in both the machine learning and speech recognition communities. Figure 1 shows a simple factorial HMM system. There are two HMMs each have three emitting states and start and end anchor points. At the anchor points, shown in black, the various streams are forced to synchronise. The factorial nature of such a model becomes clear when the total number of possible state combinations is calculated. If there are S streams (here each stream is modelled by an HMM), with N states per stream there are N^S possible state pairings. For the example in figure 1 this is 9 state combinations, but only uses the parameters from 6 states.

[1] Preferably in the sense of minimising word error rate rather than simply modelling the data .

[2] Here we are using the term factorial HMM to describe the general model in [13] rather than the specific version implemented in the paper.

[3] In this paper the term factorial will be used to denote the generic scheme in contrast to the specific factorial HMM scheme in [13].

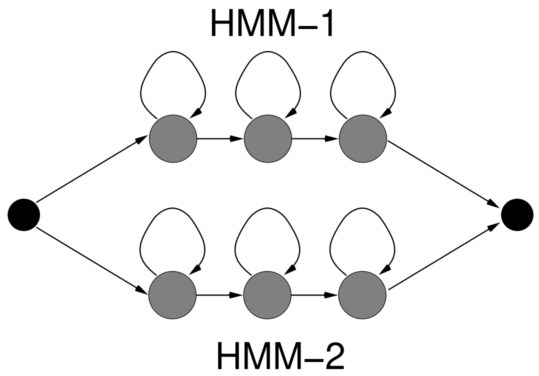

Figure 1: 2-stream factorial HMM system

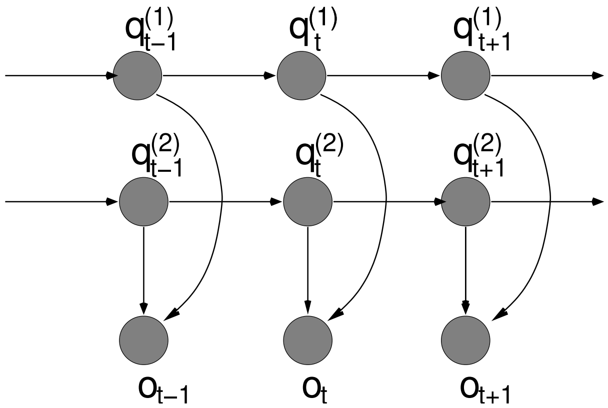

Figure 2: Directed acyclic graph for a 2-stream factorial HMM

In the machine learning community it is very popular to describe models in terms of graphs. The directed acyclic graph (DAG) associated with the factorial HMM of figure 1 is shown in figure 2. The absence of a link between nodes indicates independence. Thus the observation at time t, $\mathbf{o}_t$ is conditionally independent of all other observations given the state in stream 1, $q_t^{(1)}$, and the state in stream 2, $q_t^{(2)}$. For T observations, $\mathbf{o}_1, \ldots, \mathbf{o}_T$, the likelihood given a particular state sequence may be expressed as

$$p(\mathbf{o}_1, \ldots, \mathbf{o}_T | \mathbf{q}_1, \ldots, \mathbf{q}_T) = \prod_{t=1}^{T} p(\mathbf{o}_t | \mathbf{q}_t) \tag{1}$$

where $\mathbf{q}_t$ is the set of S states (or state-components if Gaussian mixture models (GMM)s are used at each state) that the model occupies at time t, $\left\{q_t^{(1)}, \ldots, q_t^{(S)}\right\}$. In addition to the observation conditional independence, the graph shows that the probability of being in a particular state of s stream is conditionally independent of all other streams and state positions given the previous state. Thus

$$P(\mathbf{q}_t | \mathbf{q}_{t-1}) = \prod_{s=1}^{S} P(q_t^{(s)} | q_{t-1}^{(s)}) \tag{2}$$

There are a variety of options for how the streams interact with one another to obtain the final distribution for the observation at time t.

2.1 Linear stream combination

Here both the streams, labelled *HMM-1* and *HMM-2*, generate observations in the complete feature space. The observed feature vector $\mathbf{o}_t$ is a linear combination of independent observations from each of the streams.

$$\mathbf{o}_t = \sum_{s=1}^{S} \mathbf{o}_t^{(s)} \tag{3}$$

where $\mathbf{o}_t^{(s)}$ is the observation from stream s and is assumed to be Gaussian distributed. This is the model implemented in [13]. The distributions associated with each state of each steam has a distinct mean, $\boldsymbol{\mu}^{(q^{(s)})}$, but all distributions have a common covariance matrix (i.e. $\sum_{s=1}^{S} \boldsymbol{\Sigma}^{(q_t^{(s)})} = \boldsymbol{\Sigma}, \forall t$). The likelihood of the composite, meta, state described by $\mathbf{q}_t$ generating an observation is given by

$$p(\mathbf{o}_t | \mathbf{q}_t) = \mathcal{N}(\mathbf{o}_t; \sum_{s=1}^{S} \boldsymbol{\mu}^{(q_t^{(s)})}, \boldsymbol{\Sigma}) \tag{4}$$

Re-estimation formulae and approximations for inference of such a model are given in [13]. This form of model has been successfully applied to a simple speech recognition task in [17].

A modified version of this scheme is the convolutional densities examined in [18]. The form of the topology described in the paper is a highly restricted, but relaxes the requirements that a common covariance matrix is used. Figure 3 shows the restricted topology. For the single component per state case this is identical to a single stream system. However for the multiple component per state system, here M components in "stream" 1 and K components in "stream" 2[4], the likelihood may be expressed as[5]

$$p(\mathbf{o}_t | \mathbf{q}_t) = \sum_{m=1}^{M} \sum_{k=1}^{K} w^{(m)} w^{(k)} \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}^{(m)} + \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(m)}) \tag{5}$$

[4] A simplified notation is used here. The dependence of the component on the particular state is dropped for simplicity of notation. Where this dependence is clear this simplified notation will be used.

[5] The dependence of the model parameters on the particular meta-state, $\mathbf{q}_t$, has been dropped for notational simplicity.

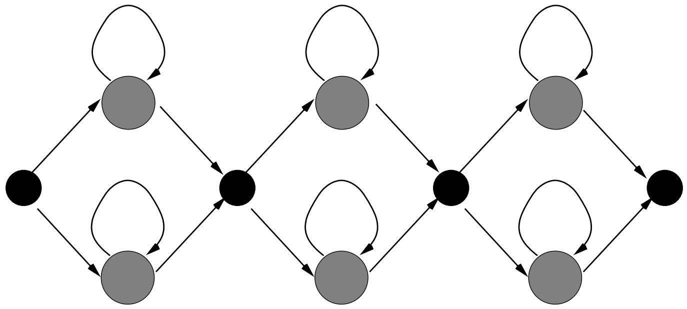

Figure 3: Restricted 2-stream factorial topology

where $q_t^{(1)} = q_t^{(2)}$ for all t. For this case there are effectively MK components, but only requiring $M + K$ model parameters to be trained. Appendix A briefly examines the training of the general factorial form of the convolutional densities. Both these forms of linear combination streams may be viewed as a simple version of the transformation stream system described in this paper.

The linear combination need not be in the same domain as the feature vector. One common form of stream combination in speech recognition is used for noise robustness [11, 27]. Here, stream 1 is used to model speech and stream 2 to model noise. The composite observation is given by[6]

$$\mathbf{o}_t = \log\left(\exp(\mathbf{o}_t^{(1)}) + \exp(\mathbf{o}_t^{(2)})\right) \tag{6}$$

In [11] this is approximated by a single Gaussian. Alternatively in [27] the observation is approximated by[7]

$$\mathbf{o}_t \approx \max\left(\mathbf{o}_t^{(1)}, \mathbf{o}_t^{(2)}\right) \tag{7}$$

The likelihood of generating the observation is given by

$$\begin{aligned} p(\mathbf{o}_t|\mathbf{q}_t) &= \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}^{(q_t^{(1)})}, \boldsymbol{\Sigma}^{(q_t^{(1)})})\mathcal{C}(\mathbf{o}_t; \boldsymbol{\mu}^{(q_t^{(2)})}, \boldsymbol{\Sigma}^{(q_t^{(2)})}) \\ &\quad + \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}^{(q_t^{(2)})}, \boldsymbol{\Sigma}^{(q_t^{(2)})})\mathcal{C}(\mathbf{o}_t; \boldsymbol{\mu}^{(q_t^{(1)})}, \boldsymbol{\Sigma}^{(q_t^{(1)})}) \end{aligned} \tag{8}$$

where $\mathcal{C}(.; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the cumulative density function for a Gaussian distribution of mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$

2.2 Independent streams

In recent years in speech recognition multiple independent stream systems, also known as multiple band systems [3, 19], have become popular. Here the two HMMs in figure 1 model distinct subsets of the feature vector. The complete feature vector may be written as

$$\mathbf{o}_t = \begin{bmatrix} \mathbf{o}_t^{(1)} \\ \vdots \\ \mathbf{o}_t^{(S)} \end{bmatrix} \tag{9}$$

[6]The observations in this section assumes that the data is modelled in the log-spectral domain. It is simple to map from the more commonly used cepstral parameters (in particular MFCCs [4]) is simple [11].

[7]The max() function here operates at a vector element level.

The complete likelihood may then be expressed as[8]

$$\begin{aligned} p(\mathbf{o}_t|\mathbf{q}_t) &= \prod_{s=1}^{S} p(\mathbf{o}_t^{(s)}|q_t^{(s)}) \\ &= \prod_{s=1}^{S} \mathcal{N}(\mathbf{o}_t^{(s)}, \boldsymbol{\mu}^{(q_t^{(s)})}, \boldsymbol{\Sigma}^{(q_t^{(s)})}) \end{aligned} \tag{10}$$

A simplified version of independent streams is described in the HTK manual [30]. The state sequences in the streams are forced to be synchronous ($q_t^{(1)} = q_t^{(2)} = \ldots = q_t^{(S)}$). The topology is the same as that of convolutional density HMMs shown in figure 3. Again, when single components per state are used there is no difference to this system and a standard HMM. However for the multiple component case the effective number of components in a state are increased rather than the number of states.

2.3 Discrete streams

In [33] the streams consist of multiple discrete values. Thus the streams are assumed to generate discrete values. The observation distribution is uniquely determined by the combination of these discrete variables. Thus[9]

$$p(\mathbf{o}_t|\mathbf{q}_t) = \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}^{(\mathbf{q}_t)}, \boldsymbol{\Sigma}^{(\mathbf{q}_t)}) \tag{11}$$

One of the major issues with this form of modelling is that the training data is partitioned into multiple distinct sets (as determined by the total number of possible combinations). This may result in problems with robustly estimating model parameters in the large complex systems typically used in speech recognition. Furthermore, it may result in large memory and runtime costs for the system.

2.4 Loosely coupled streams

Loosely coupled models [22], one form of which is the mixed memory model [26], may be viewed as a compromise between the independent stream system and and the discrete stream system. Here the distribution associated with each state of a stream is "influenced" by the states of the other streams. There are various possibilities for the nature of this influence. The extremes of the influence are the independent and discrete stream systems. In the independent stream system there is no influence on the distribution of the other streams. In contrast the discrete stream system the distribution is determined by the state of all the streams, with only the stream transitions independent of one another. Loosely coupled streams allow a compromise between the two systems to be made (and, in a more general model than the factorial HMM, coupling for the stream transitions).

The mixed memory model [26] uses the following method for describing the "influence" of the emission probabilities of one stream on another

$$p(\mathbf{o}_t|\mathbf{q}_t) = \prod_{s=1}^{S} \left(\sum_{u=1}^{S} \lambda_u^{(s)} p(\mathbf{o}_t^{(s)}|q_t^{(u)}) \right) \tag{12}$$

where the stream weights, $\lambda_u^{(s)}$, satisfy

$$\sum_{u=1}^{S} \lambda_u^{(s)} = 1; \quad \lambda_u^{(s)} \geq 0 \tag{13}$$

[8]It is interesting to note that the factorial HMM system [13] may be viewed as subsuming this expression by appropriately zeroing the mean parameters of each of the streams.

[9]The equation here shows only a single Gaussian component for the observation state. For the general case a mixture of any distributions may be used.

The independent steam case is simply described by

$$\lambda_u^{(s)} = \begin{cases} 1, & u = s \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

Hence in addition to the state distributions a set of S^2 stream weights are also trained. The advantage of such a factorisation over a straight meta-state model becomes clear when the number of model parameters is considered. For simplicity consider the case where the d-dimensional feature vector $\mathbf{o}_t$ is equally partitioned into S streams and there are N states for all stream models. For a complete meta-state model there are $\mathcal{O}(dN^S)$ parameters (assuming the distribution has parameters $\mathcal{O}(d)$, though this is not the case for full covariance matrices). For the mixed memory model there are $\mathcal{O}(dNS + S^2)$ parameters. However, comparing the number of model parameters with that of the independent stream system, $\mathcal{O}(dN)$, shows that there is a, possibly large, increase in the number of model parameters over an independent stream system. An alternative to mixed memory model is a generalised version of parameter tying [21].

Though to date the application of these models to speech recognition has been unsuccessful [22], it is an interesting extension to the standard factorial HMM framework.

3 Transformation streams

In speech recognition the use of linear transformations of both the model parameters [16, 8] and feature vectors [8, 9] are very popular. One option in the factorial HMM system is use transformations at each of the states of one stream to transform the features or model parameters of another stream. In a similar fashion to loosely coupled streams this may be seen as an approximation to the discrete stream system without the dramatic increase in the number of model parameters. Consider a simple two valued discrete stream representing speaker style, for example one "fast" and the other "slow". Rather than using a discrete stream to partition the training data (possibly in a probabilistic fashion) a transform (or more generally set of transforms) are associated with each value. In common with speaker adaptation this allows a system to be "adapted" in this case to speaker style with very few parameters. It is possible to train such systems in the same fashion as adaptive training schemes described in [1, 8].

There are a variety of transforms that may be used. Other than the generalisation of the linear stream combination system, only 2 stream systems are described. One of the streams has a conventional set of distributions (in this case stream 2) associated with the states. The other stream a set of transformations.

- **Interpolation weights**: The simplest transformation is to use a set of interpolation weights when summing the stream means, rather than using the straight summation of the linear stream combination. Here

$$p(\mathbf{o}_t|\mathbf{q}_t) = \mathcal{N}(\mathbf{o}_t; \sum_{s=2}^{S} \lambda_s^{(q_t^{(1)})} \boldsymbol{\mu}^{(q_t^{(s)})}, \boldsymbol{\Sigma}^{(q_t^{(S)})}) \tag{15}$$

  where $\lambda_s^{(q^{(1)})}$ is the stream interpolation weight of stream s given that stream 1 (the transformation stream) is in state $q^{(1)}$. The estimation of the model parameters is a simple generalisation of the cluster adaptive training scheme described in [10].

- **Maximum likelihood linear regression** (MLLR) [16]: Here a linear transformation of Gaussian component means is used to transform the distributions. In this case

$$p(\mathbf{o}_t|\mathbf{q}_t) = \mathcal{N}(\mathbf{o}_t; \mathbf{A}^{(q_t^{(1)})} \boldsymbol{\mu}^{(q_t^{(2)})} + \mathbf{b}^{(q_t^{(1)})}, \boldsymbol{\Sigma}^{(q_t^{(2)})}) \tag{16}$$

  where $\mathbf{A}^{(q^{(2)})}$ and $\mathbf{b}^{(q^{(2)})}$ are the matrix transformation and bias vector associated with state $q^{(2)}$. Adaptive training for such a transform is described in [1]. However for large systems with large model sets this becomes computationally (and memory) intensive. In addition to adapting the means it is possible to adapt the variances.

- **Constrained MLLR** (or feature-space transformation): Rather than adapting the model parameters the feature vector may be transformed. In this case the likelihood is expressed as

$$p(\mathbf{o}_t|\mathbf{q}_t) = |\det(\mathbf{A}^{(q_t^{(1)})})|\mathcal{N}(\mathbf{A}^{(q_t^{(1)})}\mathbf{o}_t + \mathbf{b}^{(q_t^{(1)})}; \boldsymbol{\mu}^{(q_t^{(2)})}, \boldsymbol{\Sigma}^{(q_t^{(2)})}) \tag{17}$$

  The advantage of using a transformation of the features is that the adaptive training of the model parameters is simple [8] requiring minimal changes to the standard training schemes. The linear stream combination systems may be viewed as constrained versions of this transform (and the MLLR transforms). The means from one stream become transform biasses. Though bias transforms have been used for adaptation of simple transforms have been shown to be less effective than more complex ones [20]. This suggests that the use of these more complex transformation streams may be useful. Moreover the use of the linear transform allows data generated from different sources to be more effectively normalised (see section 5 for more details).

- **Non-linear transformations**: It is not necessary to restrict the transforms to be linear. For the non-linear model transformation case

$$p(\mathbf{o}_t|\mathbf{q}_t) = \mathcal{N}(\mathbf{o}_t; \mathcal{F}_{\mu}^{(q_t^{(1)})}(\boldsymbol{\mu}^{(q_t^{(2)})}, \boldsymbol{\Sigma}^{(q_t^{(2)})}), \mathcal{F}_{\Sigma}^{(q_t^{(1)})}(\boldsymbol{\mu}^{(q_t^{(2)})}, \boldsymbol{\Sigma}^{(q_t^{(2)})})) \tag{18}$$

  However, if non-linear transformations are to be used then there are typically no simple formulae to estimate the model parameters (nor the transform parameters).

As an alternative to associating a non-linear transformation with each stream state, a mixture of linear transformations may be used. Each transform has an associated component weight, $w^{(m)}$. For the simplest case of a single state, M-component, constrained MLLR transformation stream, the likelihood of state $q_t^{(2)}$ generating observation $\mathbf{o}_t$ is given by

$$p(\mathbf{o}_t|q_t^{(2)}) = \sum_{m=1}^{M} w^{(m)}|\det(\mathbf{A}^{(m)})|\mathcal{N}(\mathbf{A}^{(m)}\mathbf{o}_t + \mathbf{b}^{(m)}; \boldsymbol{\mu}^{(q_t^{(2)})}, \boldsymbol{\Sigma}^{(q_t^{(2)})}) \tag{19}$$

A simple form of mixture of transformations has previously been used for speaker adaptation in [6] and as an extension to factor analysis [12]. By using more than one transformation component a non-linear state-stream specific transformation may be obtained.

The HMM error model (HEM) sits within this class of factorial model. The transformation stream used is a constrained MLLR transform. This has two important consequences. First for large systems it is expensive to adapt the model parameters when using model-space transformations. Second updating the parameters of the other streams given the transformation is simple in this case [8].

4 Multiple Transformation Streams

It is interesting to examine the form of the model when there are multiple transformation streams. Figure 4 shows a three stream system where streams 1 and 2 are both feature-space transformation streams and stream 3 is an HMM. Using the factorial HMM notation where $\mathbf{q}_t$ indicates the meta-state occupancy at time t and only considering a single component per state[10] the likelihood of a particular observation is given by

$$p(\mathbf{o}_t|\mathbf{q}_t) = \left(\prod_{s=1}^{S-1} |\det(\mathbf{A}^{(q_t^{(s)})})|\right) \mathcal{N}\left(\mathbf{o}_t^{(S-1)}; \boldsymbol{\mu}^{(q_t^{(S)})}, \boldsymbol{\Sigma}^{(q_t^{(S)})}\right) \tag{20}$$

[10]The extension to multiple components is trivial.

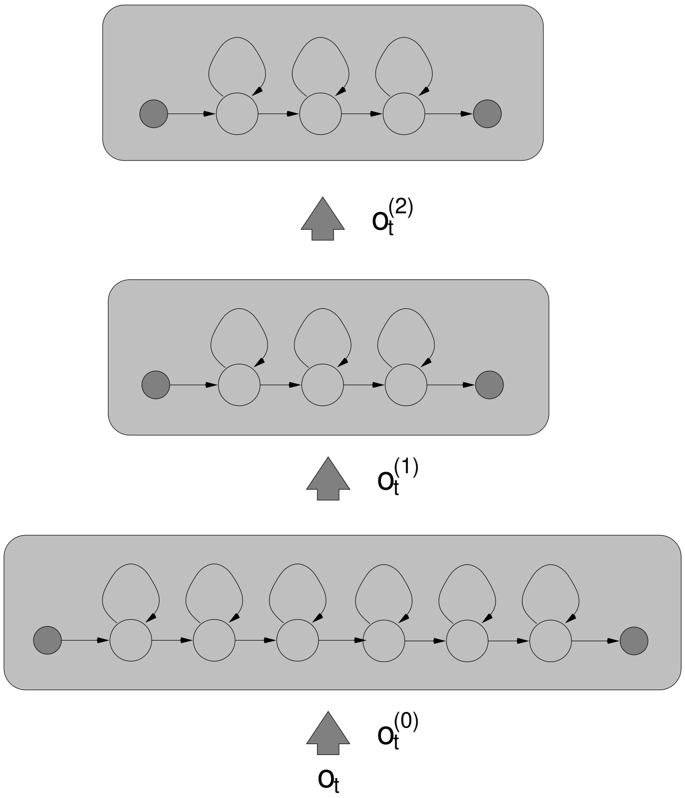

Figure 4: Multiple Transformation Streams

where the following recursion is used to obtain the transformed observation

$$\mathbf{o}_t^{(s)} = \mathbf{W}^{(q_t^{(s)})} \begin{bmatrix} \mathbf{o}_t^{(s-1)} \\ 1 \end{bmatrix} = \mathbf{W}^{(q_t^{(s)})} \boldsymbol{\zeta}_t^{(s-1)} \tag{21}$$

where $\mathbf{o}_t^{(0)} = \mathbf{o}_t$ and

$$\mathbf{W}^{(q^{(s)})} = \begin{bmatrix} \mathbf{A}^{(q^{(s)})} & \mathbf{b}^{(q^{(s)})} \end{bmatrix} \tag{22}$$

The model parameters may again be estimated in an EM fashion. The details of the optimisation are given in Appendix B.

An interesting modification to the multiple transformation stream system described above is to allow the dimensionality of the transformations/models associated with each of the stream levels to vary. Consider the simple two stream case, where the dimensions associated with stream s is $d^{(s)}$. In order to ensure that likelihoods may be compared across models (which may possibly have different dimensions associated with them), the two stream case likelihood is written as

$$p(\mathbf{o}_t|\mathbf{q}_t) = \mathcal{N}(\mathbf{o}_t; \mathbf{A}^{(q_t^{(1)})}\boldsymbol{\mu}^{(q_t^{(2)})} + \boldsymbol{\mu}^{(q_t^{(1)})}; \mathbf{A}^{(q_t^{(1)})}\boldsymbol{\Sigma}^{(q_t^{(2)})}\mathbf{A}^{(q_t^{(1)})T} + \boldsymbol{\Sigma}^{(q_t^{(1)})}) \tag{23}$$

This has the following general generative model[11]

$$\mathbf{o}_t^{(s-1)} = \mathbf{A}^{(q_t^{(s)})}\mathbf{o}_t^{(s)} + \mathbf{v}_t^{(s)} \tag{24}$$

where

$$\mathbf{v}_t^{(s)} \sim \mathcal{N}(\boldsymbol{\mu}^{(q_t^{(s)})}, \boldsymbol{\Sigma}^{(q_t^{(s)})}) \tag{25}$$

[11] For simplicity only single components are used in the equations. In the general case mixtures are used.

where $\mathbf{A}^{(q_t^{(s)})}$ is an $d^{(s)} \times d^{(s+1)}$ matrix, $\mathbf{o}_t^{(S)} = \mathbf{v}_t^{(S)}$, and again $\mathbf{o}_t^{(0)} = \mathbf{o}_t$. This is a generalisation of factor analysis [25], mixture of factor analysers [12] and shared factor analysis [14]. This form of model will be investigated in future work.

5 HMM Error Model

The model examined in this paper, the HMM error model (HEM), is a 2-stream factorial model using a feature-space transformation stream. Furthermore the second stream model is restricted to having only a single emitting state (effectively a GMM). As such it is not make full use of the power of a factorial HMM system, but does sit within the general class of factorial HMMs. A HEM model, with a three state transformation stream, is shown in figure 5.

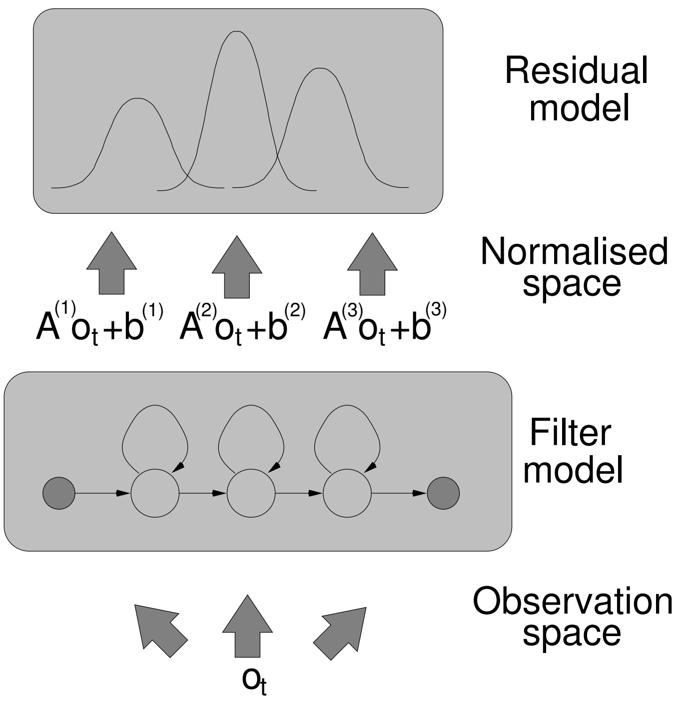

Figure 5: HMM Error Model

Due to the relatively simple model used there is another intuitive interpretation of the HEM. Rather than considering it as a multiple stream system it could also described in terms of a filter model and residual model. Thus the first stream, a transformation stream, is a filter model. The second stream models the residual from the filter model. The aim of the filter model is to transform the data from the observation space into some normalised space such that the data is independent and identically distributed. For this work a mixture of linear feature-space transformations is associated with each state of the filter model. If the filter model is initialised using HMMs trained with ML estimation, the data in the normalised space will have zero mean and an identity matrix covariance matrix. However the higher order statistics are not constrained in anyway. Furthermore, for speech recognition diagonal covariance matrices are commonly used. In this case the leading diagonal of the covariance matrix is constrained to consist of ones. Hence by using either a GMM, or more complex covariance matrix Gaussian distribution, as the residual model, it is possible to more accurately model the data distribution in the normalised space.

A model similar to the HEM has previously been investigated for variance compensation in speaker adaptation [8]. The filter model was constrained to have diagonal transforms and the residual model a single speaker-specific full covariance matrix multi-variate Gaussian distribution. This allowed feature vector correlations to be modelled. The model described in [8] differs from the

HEM in a couple of ways. First the residual model is trained per speaker. The HEM residual model is trained on all the training data (though may, of course, be adapted when speaker adaptation is being used). Second the filter model was not adaptively trained, the transform parameters were set using the initialisation scheme described later in section 5.3.1.

This section describes the likelihood calculation for a HEM model and how the model may be trained using ML estimation. In addition some implementation issues are addressed.

5.1 Likelihood Calculation

From the description of the HEM model in figure 5 the likelihood is calculated by computing the likelihood of the normalised observations using the residual model. However appropriate scaling is required to account for the component weights of the filter model and the effects of the feature transformation of the filter model. Consider a single state q of an HMM modelled with M Gaussian components and a K-component GMM residual model. The likelihood of the state q may be written as

$$p(\mathbf{o}_t|q_t = q) = \sum_{m=1}^{M}\sum_{k=1}^{K} w^{(m)}w^{(k)}|\det(\mathbf{A}^{(m)})|\mathcal{N}(\mathbf{W}^{(m)}\boldsymbol{\zeta}_t; \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)}) \tag{26}$$

where state q has M components, $1, \dots, M$,

$$\mathbf{W}^{(m)} = \begin{bmatrix} \mathbf{A}^{(m)} & \mathbf{b}^{(m)} \end{bmatrix} \tag{27}$$

$$\boldsymbol{\zeta}_t = \begin{bmatrix} \mathbf{o}_t \\ 1 \end{bmatrix} \tag{28}$$

Various forms of transformation matrix $\mathbf{A}^{(m)}$ may be used. These range from simple diagonal transforms to full matrix transforms. The most common form of covariance model used in speech recognition is diagonal. In terms of viewing the HEM as a filter and residual model this corresponds to a diagonal transform. This will be the form of model primarily discussed.

The likelihood may also be expressed in a more standard form using

$$p(\mathbf{o}_t|q_t = q) = \sum_{i=1}^{MK} w^{(i)}\mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}^{(i)}, \boldsymbol{\Sigma}^{(i)}) \tag{29}$$

where

$$w^{(i)} = w^{(m)}w^{(k)} \tag{30}$$

$$\boldsymbol{\mu}^{(i)} = \mathbf{A}^{(m)-1}(\boldsymbol{\mu}^{(k)} - \mathbf{b}^{(m)}) \tag{31}$$

$$\boldsymbol{\Sigma}^{(i)} = \mathbf{A}^{(m)-1}\boldsymbol{\Sigma}^{(k)}\mathbf{A}^{(m)T-1} \tag{32}$$

and m and k correspond to the expanded space component i. So in the standard factorial fashion using $M + K$ components MK components are generated.

5.2 Parameter Estimation

This section describes how both the filter model and the residual model may be trained using expectation maximisation (EM) [5]. As with other factorial HMM schemes multiple levels of indicator variables are required. The likelihood of the HEM, $\mathcal{M}$, generating an observation sequence $\mathbf{O} = \mathbf{o}_1, \dots, \mathbf{o}_T$ is given by

$$\mathcal{L}(\mathbf{O}; \mathcal{M}) = \sum_{\boldsymbol{\Theta}}\prod_{t=1}^{T}\left(p(q_t|q_{t-1})\sum_{m\in\theta(\tau)}\sum_{k=1}^{K} w^{(m)}w^{(k)}|\det(\mathbf{A}^{(m)})|\mathcal{N}(\mathbf{W}^{(m)}\boldsymbol{\zeta}_t; \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)})\right) \tag{33}$$

where $\boldsymbol{\Theta}$ is the set of all valid state sequences according to the transcription for the data, q_t is the state at time t of the current path and $\theta(t)$ is set of Gaussian components belonging to state. It is simple to show that the following auxiliary function is obtained

$$\begin{aligned} \mathcal{Q}(\mathcal{M},\hat{\mathcal{M}}) &= \sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{t=1}^{T}\gamma_t^{(mk)}\left(\log(w^{(m)})+\log(w^{(k)})+\log(|\det(\mathbf{A}^{(m)})|)\right. \\ &\left.-\frac{1}{2}\left(\log(\boldsymbol{\Sigma}^{(k)})+(\mathbf{W}^{(m)}\boldsymbol{\zeta}_t-\boldsymbol{\mu}^{(k)})\boldsymbol{\Sigma}^{(k)-1}(\mathbf{W}^{(m)}\boldsymbol{\zeta}_t-\boldsymbol{\mu}^{(k)})\right)\right) \end{aligned} \tag{34}$$

where $\gamma_t^{(mk)}$ is the probability of being in component m of the filter model and residual component k at time t and M is the *total* number of transform components in the filter model. To simultaneously update both the filter model parameters and residual model parameters is highly complex (and for most tasks impractical). Instead a simple iterative maximisation scheme is used. First the filter model parameters are estimated given the current estimates of the residual model. Second the residual model parameters are updated given the filter model. The two optimisation schemes are described below.

5.2.1 Filter Model Estimation

The filter model parameters, $w^{(m)}$, $\mathbf{W}^{(m)}$ for each component m of the filter must be estimated given the current estimate of the residual model parameters. In the trivial case where the residual model is a single diagonal covariance matrix Gaussian component the filter model parameter estimation is identical to the standard HMM estimation schemes (see the initialisation section for an interpretation of the model parameters). For the more interesting GMM residual model case, the form of equation 34 is very similar to the estimation of a constrained MLLR transform given in [8]. Only the case where the covariance matrices of the residual components are diagonal is considered[12]. The ML estimate of the i^{th} row of $\mathbf{W}^{(m)}$, $\mathbf{w}_i^{(m)}$, may be shown to be

$$\mathbf{w}_i^{(m)} = \left(\alpha\mathbf{p}_i+\mathbf{k}^{(i)}\right)\mathbf{G}^{(i)-1} \tag{35}$$

where $\mathbf{p}_i$ is the extended cofactor row vector $\begin{bmatrix} p_{i1} & \dots & p_{in} & 0 \end{bmatrix}$, ($p_{ij} = \text{cof}(\mathbf{A}_{ij}^{(m)})$),

$$\mathbf{G}^{(i)} = \sum_{k=1}^{K}\frac{1}{\sigma_i^{(k)2}}\sum_{t=1}^{T}\gamma_t^{(mk)}\boldsymbol{\zeta}_t\boldsymbol{\zeta}_t^T \tag{36}$$

$$\mathbf{k}^{(i)} = \sum_{k=1}^{K}\frac{1}{\sigma_i^{(k)2}}\mu_i^{(k)}\sum_{t=1}^{T}\gamma_t^{(mk)}\boldsymbol{\zeta}_t^T \tag{37}$$

and α satisfies

$$\alpha^2\mathbf{p}_i\mathbf{G}^{(i)-1}\mathbf{p}_i^T+\alpha\mathbf{p}_i\mathbf{G}^{(i)-1}\mathbf{k}^{(i)T}-\left(\sum_{k=1}^{K}\sum_{t=1}^{T}\gamma_t^{(mk)}\right)=0 \tag{38}$$

The optimisation scheme is iterative since the estimation of each matrix row is influenced by the cofactors of the complete matrix. For the simple case where the transformation matrix $\mathbf{A}^{(m)}$ is diagonal, the cofactors are zero so a closed form solution is possible.

In addition to the transform parameters the weights of the components of the filter model are required. Comparing equation 34 to the standard HMM re-estimation formulae shows that

$$w^{(m)} = \frac{\sum_{k=1}^{K}\sum_{t=1}^{T}\gamma_t^{(mk)}}{\sum_{i\in\mathcal{J}^{(m)}}\sum_{k=1}^{K}\sum_{t=1}^{T}\gamma_t^{(ik)}} \tag{39}$$

where $\mathcal{J}^{(m)}$ is the set of transform components belonging to the same filter model state as component m.

[12] The more general full covariance case is a simple modification to the training given in [7].

5.2.2 Residual Model Estimation

Once the filter model has been trained the estimation of the residual model is very similar to the standard GMM training. It is simple to show that the mean of residual component k is given by

$$\boldsymbol{\mu}^{(k)} = \frac{\sum_{m=1}^{M}\sum_{t=1}^{T}\gamma_t^{(mk)}\mathbf{W}^{(m)}\boldsymbol{\zeta}_t}{\sum_{m=1}^{M}\sum_{t=1}^{T}\gamma_t^{(mk)}} \tag{40}$$

and similarly for the covariance matrix

$$\boldsymbol{\Sigma}^{(k)} = \frac{\sum_{m=1}^{M}\sum_{t=1}^{T}\gamma_t^{(mk)}(\mathbf{W}^{(m)}\boldsymbol{\zeta}_t - \boldsymbol{\mu}^{(k)})(\mathbf{W}^{(m)}\boldsymbol{\zeta}_t - \boldsymbol{\mu}^{(k)})^T}{\sum_{m=1}^{M}\sum_{t=1}^{T}\gamma_t^{(mk)}} \tag{41}$$

The estimation of residual component weights follows the standard GMM optimisation. Thus

$$w^{(k)} = \frac{\sum_{m=1}^{M}\sum_{t=1}^{T}\gamma_t^{(mk)}}{\sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{t=1}^{T}\gamma_t^{(mk)}} \tag{42}$$

5.3 Implementation Issues

5.3.1 Parameter Initialisation

In common with other EM-based schemes it is important to have reasonable estimates to initialise the training. The initialisation of the HEM model parameters has two distinct stages. First the filter model parameters are estimated assuming that a single, zero mean, identity covariance matrix Gaussian residual model is used. As previously mentioned this simply requires training a standard HMM. The conversion of the standard HMM parameters into the filter model parameters is achieved using

$$\mathbf{A}^{(m)} = \mathbf{C}^{(m)-1} \tag{43}$$

$$\mathbf{b}^{(m)} = -\mathbf{A}^{(m)}\boldsymbol{\mu}^{(m)} \tag{44}$$

where

$$\mathbf{C}^{(m)}\mathbf{C}^{(m)T} = \boldsymbol{\Sigma}^{(m)} \tag{45}$$

Note that in equation 26 the "transform" $\mathbf{A}^{(m)}$ is a full matrix. However from the standard full covariance HMM it can be initialised with an upper triangular matrix from the Choleski factorisation of the covariance matrix. The need for a full transformation matrix when there is more than one component in the residual GMM (i.e the set of components K), is the same as the need for full semi-tied transforms [9].

Once the filter model parameters have been estimated, the residual model can be initialised using any standard scheme for initialising GMMs. For the work presented here a scheme similar to the HTK *mixing-up* routine was used [30] (see section 5.3.4 for more details).

5.3.2 Flooring

All state-of-the-art HMM-based speech recognition systems apply a variance floor in the estimation stage (see for example [30]). It is interesting to see how variance flooring may be applied to HEM models. There are two places that flooring may be applied. The first place is the residual model variances . This is straight-forward particularly as the variances of the residual model are expected to be around one. The more interesting aspects of the flooring is how to appropriately floor the filter model. For simplicity only the case of diagonal covariance matrices (which result in diagonal transforms). For this case the transform parameter estimation is non-iterative. By analogy with the standard variance flooring the *maximum* value that a_i can take is $1/f_i$ where f_i is the floor value for the i^{th} element. Using equation 35 $\mathbf{w}_i$, hence a_i, may be found. If a_i exceeds $1/f_i$ it is then set to $1/f_i$. It is now necessary to find the ML estimate of b_i. Setting the value of a_i to f_i then (note that for the diagonal transform case $\mathbf{G}^{(i)}$ is a 2×2 matrix and $\mathbf{k}^{(i)}$ is a 2-dimensional vector) yields

$$\begin{aligned} b_i^{(m)} &= \frac{\sum_{k=1}^{K}\sum_{t=1}^{T}\frac{\gamma_t^{(mk)}}{\sigma_i^{(k)2}}(o_{ti}/f_i-\mu_i^{(k)})}{\sum_{k=1}^{K}\sum_{t=1}^{T}\frac{\gamma_t^{(mk)}}{\sigma_i^{(k)2}}} \\ &= \frac{g_{12}^{(i)}/f_i-k_2^{(i)}}{g_{22}^{(i)}} \end{aligned} \tag{46}$$

It is interesting to see that the ML estimate of the bias-equivalent term is altered if flooring is applied. This is not the case for standard HMMs or GMMs.

When full transformations are used the simplest approach is to set a maximum value on the leading diagonal elements of $\mathbf{A}^{(m)}\mathbf{A}^{(m)T}$. This is similar to the flooring scheme used for full-covariance matrix systems in HTK [30].

5.3.3 Memory and Computational Cost

Two important issues in speech recognition (and many other applications) are the memory and runtime computational cost of the system. For the HEM system there is a minimal increase in the number of model parameters. The typical large vocabulary speech recognition system may have over 100,000 diagonal covariance matrix Gaussian components. For the experiments in this paper only 3 additional components are required[13].

The computational cost of calculating the likelihoods with a HEM system are significantly greater than that of a standard HMM system of comparable complexity to the filter model. Since the HEM model results in a GMM (of the complexity of the residual model) being generated for every component of the filter model the runtime likelihood calculation cost scales linearly with the number of components in the residual model. However by using schemes such as Gaussian selection [2] the cost of the likelihood calculation compared to the search may be dramatically reduced.

5.3.4 Mixing Up

The results presented in this paper are based on a system built using HTK [30]. In order to build the system mixing up[14] was used. The residual model is a standard GMM which may be mixed-up in the standard HTK fashion of perturbing the means. However, since we are building mixtures of transforms, rather than components, for the filter model, it is interesting to consider how to mix-up a mixture of transforms. The likelihood of a particular M-component filter model state, q, and K-component residual model pairing may be written as

$$\begin{aligned} p(\mathbf{o}_t|q_t=q) &= \sum_{m=1}^{M}\sum_{k=1}^{K} w^{(m)}w^{(k)}|\det(\mathbf{A}^{(m)})|\mathcal{N}(\mathbf{W}^{(m)}\mathbf{o}_t;\boldsymbol{\mu}^{(k)},\boldsymbol{\Sigma}^{(k)}) \\ &= \sum_{m=1}^{M}\sum_{k=1}^{K} w^{(m)}w^{(k)}|\det(\mathbf{A}^{(m)})|\mathcal{N}(\mathbf{A}^{(m)}\mathbf{o}_t;\boldsymbol{\mu}^{(k)}-\mathbf{b}^{(m)},\boldsymbol{\Sigma}^{(k)}) \end{aligned} \tag{47}$$

where components 1 to M are associated with state q. So to perturb the mean of the "meta-component" involves perturbing only the transformation bias. In addition it is assume that the

[13] If full transforms are used, equivalent to initialising the filter model with a full covariance matrix system, there is an increase, almost a doubling, in the number of model.

[14] Mixing-up involves gradually increasing the number of Gaussian components in a particular state. The standard procedure is to take the Gaussian component with the largest weight, perturb the means to generate two components and retrain the system.

average variance in the residual model is approximately 1 (the residual model is approximately initialised to this range). The perturbation operation is then

$$\left[\begin{array}{cc}\mathbf{A}^{(m)} & \mathbf{b}^{(m)}\end{array}\right] \rightarrow \left\{\begin{array}{l}\left[\begin{array}{cc}\mathbf{A}^{(m)} & (\mathbf{b}^{(m)}+\epsilon)\end{array}\right] \\ \left[\begin{array}{cc}\mathbf{A}^{(m)} & (\mathbf{b}^{(m)}-\epsilon)\end{array}\right]\end{array}\right. \tag{48}$$

where ϵ is the perturbation value. For the experiments used in this paper ϵ is set 0.2. The component weight is evenly divided between the two new transforms.

6 Results

The performance of the HEM models was evaluated on two standard large-vocabulary speaker-independent speech recognition tasks. The first, Hub1 Wall Street Journal (WSJ), is a scripted speech database where speakers were asked to read passages from the WSJ. The second task, Hub5, is a telephone bandwidth spontaneous speech recognition task. For all the experiments presented here diagonal covariance matrices are used for all Gaussian components (including the residual model where used). The transforms of the filter model were also constrained to be diagonal unless otherwise stated. The same decision tree clustering was used for determining the context dependent standard HMM states and the HEM transformation states. Hence the number of model parameters for a standard HMM system and a diagonal transform HEM system are about the same for the same number of components in the filter model. The filter model transform parameters were initialised with a single component standard HMM system as described in 5.3.1. The three component residual model was initialised to means at 0.2, 0.0 and -0.2 for each dimension and identity covariance matrices.

6.1 Wall Street Journal Experiments

The baseline system used for the WSJ recognition task was a gender-independent cross-word-triphone mixture-Gaussian tied-state HMM system. This was the same as the “HMM-1” model set used in the HTK 1994 ARPA evaluation system [28]. In this model set all the speech models had a three emitting state, left-to-right topology. Two silence models were used. The first silence model, a short pause model, had a single emitting state which may be skipped. This model was used to represent short inter-word silences. The other silence model was a fully connected three emitting state model used to represent longer periods of silence. The speech was parameterised into 12 MFCCs, C_1 to C_{12}, along with normalised log-energy and the first and second differentials of these parameters. This yielded a 39-dimensional feature vector, to which cepstral mean normalisation was applied. The acoustic training data consisted of 36493 sentences from the SI-284 WSJ0 and WSJ1 sets, and the LIMSI 1993 WSJ lexicon and phone set were used. The standard HTK system was trained using decision-tree-based state clustering [31] to define 6399 speech states. For the H1 task a 65k word list and dictionary was used with the trigram language model described in [28]. All decoding used a dynamic-network decoder [23].

When generating the multiple component systems used for this task, mixing-up was used [30]. The performance was investigated at various stages of this process. It should be emphasised that the grammar scale factor and insertion penalties were not optimised at any stage for the particular number of components in the system. A three Gaussian component residual model was used.

The average performance on the WSJ 1994 development and evaluation data is shown in figure 6 for a standard system and a 3-component HEM system at various numbers of components per state. For all size systems the HEM models outperformed the standard systems. The 12 component HEM system was 4% relative better than the standard system. This was significantly better at a 95% confidence using a pair-wise significance test. It is also interesting to note that the 8-component HEM system outperforms the 12-component standard system.

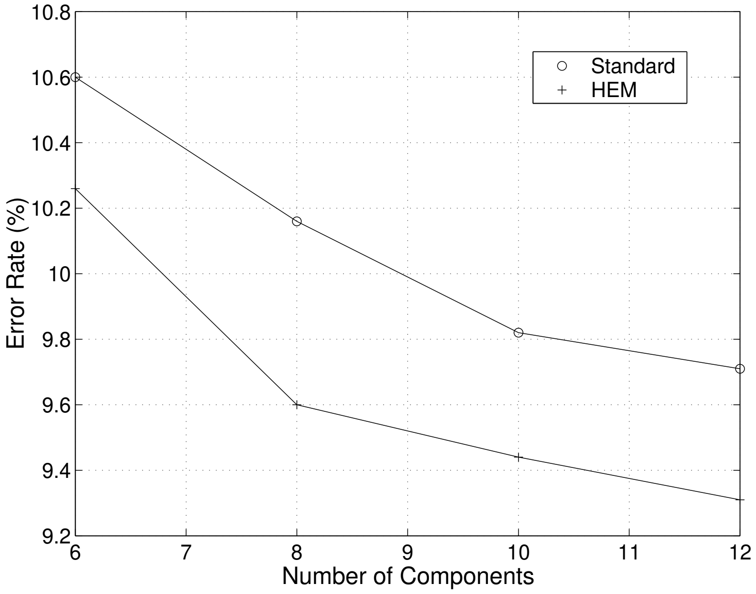

Figure 6: Average performance on WSJ 1994 H1 development and evaluation data

6.2 Switchboard Experiments

The Switchboard (Hub5) acoustic training data is obtained from two corpora: Switchboard-1 (Swb1) and Call Home English (CHE). The full training corpus consists of 180 hour training set, 4482 sides from Swb1 and 235 sides from CHE. For the experiments performed in this section a subset of this was used. A total of 68 hours was chosen to include all the speakers from Swb1 in h5train00 as well as a subset of the available CHE sides. 862 Swb1 sides and 92 CHE sides were used in this subset. This is the "h5train00sub" training set described in [15]. The speech waveforms were coded using perceptual linear prediction cepstral coefficients derived from a Mel-scale filterbank (MF-PLP) covering the frequency range from 125Hz to 3.8kHz. A total of 13 coefficients, including c_0, and their first and second order derivatives were used. Cepstral mean subtraction and variance normalisation were performed for each conversation side. Vocal tract length normalisation (VTLN) was applied in both training and test. In common with the Hub1 task a gender-independent cross-word-triphone mixture-Gaussian tied-state HMM system was built

The model sets generated were trained in the standard HTK fashion using mixing-up. State-based decision tree clustering was used to define a total of 6165 distinct speech states. Again the performance of the system was examined at various stages of the mixing up process. For all experiments a trigram language model was used, built as in [15]. A three Gaussian component residual model was again used. In addition to a global residual model, phone specific residual models were also built.

The performance of a standard system, a global three-component-residual-model HEM system and a phone-based, three-component-residual-model, HEM system are shown in figure 7. The performance of the global HEM residual model was consistently better than the standard HMM system. The twelve component system performance of 46.1% was significantly better at the 95% level than the standard system. The phone residual model performance was consistently slightly better than the global residual model system. Again it is interesting to note that the 8-component phone level residual model HEM system out performs the 12-component standard system. The number of components of the standard system was increased to 14. This increased the error rate to 46.8%, indicating that simply increasing the number of components in a standard system improves performance on this task.

The absolute gain obtained using the phone-based HEM models was small (0.9% absolute),

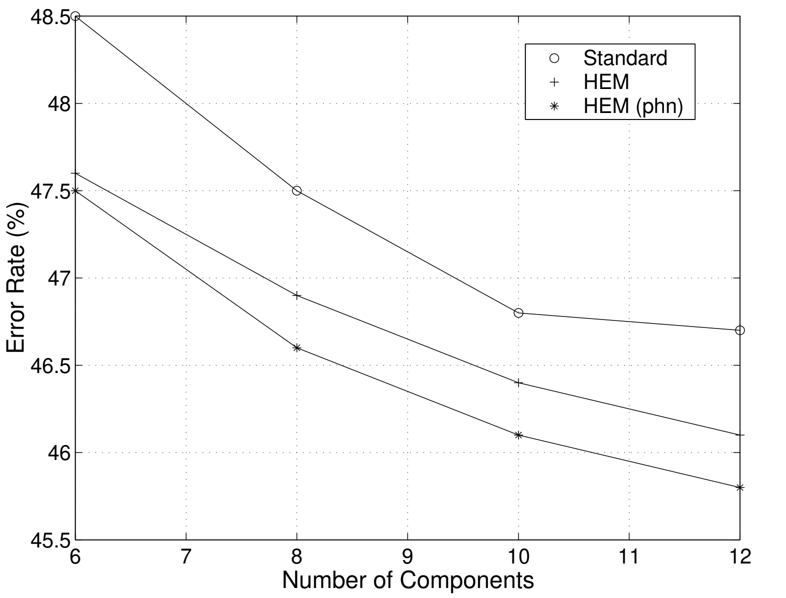

Figure 7: 1998 Switchboard evaluation performance

though it was significant at the 95% confidence level. The performance of the HEM systems may be compared with other soft tying schemes[15] An implementation of the state soft-tying scheme as described in [15], using the same training data, gave an error rate of 46.2% compared to the 45.8% obtained using a phone level HEM[16]. One interpretation of the the residual model is that it is, crudely, modelling the correlations of the feature vector. A standard soft tying approach to this is to use semi-tied covariance matrices [9]. On this task a global semi-tied transform yielded an error rate of 46.1%. Thus the use of a simple global residual model achieves about the same performance as a global semi-tied system on this task.

7 Conclusions

This paper has introduced a new form of factorial model stream, the transformation stream. This new form of stream is shown to generalise the standard factorial HMM and the convolutional densities investigated in speech recognition. A particular form of factorial HMM, the HMM error model, was then described. This model may also be described in terms of a non-linear filter and residual model. The filter model transforms the original set of feature vectors into a space in which the data should be zero mean and identity covariance matrix. However due to inaccuracies in the model, for example if diagonal covariance matrices are used at the base level, off-diagonal terms may exist in the normalised and non-zero higher order moments may occur, it is necessary to use non-Gaussian residual models, in this case a GMM. This form of model was evaluated on two large vocabulary speech recognition tasks, one involving read speech the other spontaneous telephone speech. On both test sets the new form of model performed significantly better than standard HMMs.

Very few of the possible options for the HEM model, or more generally the use of transformation streams, have been investigated in this paper. In particular the form of the HEM model used here

[15]Due to the nature of the HEM, which uses a GMM for the residual model, it is more appropriate to compare the performance with soft tied systems rather than factorial HMM systems. To the author's knowledge there are no performance figures for factorial HMMs on systems of this size.

[16]In the published paper [15] the soft results quote on the larger 180 hour training data set. It should also be noted that due to minor differences in the initialisation of the standard systems the baseline performance for the h5train00sub training was 46.6%, compared to the 46.7% obtained in these experiments.

concentrates the complexity on the filter model rather than for example using a more complex residual model (possibly using the full power of factorial HMMs with transformation streams). The model has also not been assessed in terms of how it performs with speaker adaptation (nor whether it is better to simply adapt the residual model rather than the filter model). Finally the use of discriminative training techniques is popular in state-of-the-art speech recognition [29]. It is interesting to know whether HEMs are suitable for discriminative training.

Acknowledgements

Some of the original ideas for the HEM model resulted from discussions with Dr S. Chen of Renaiassance Technology. The author gratefully acknowledges the use of equipment supplied by IBM under an SUR award.

A Parameter Estimation with a Bias Transform

This appendix describes the generalisation of the standard factorial HMM scheme [13] to the case where the covariance matrices are not constrained to be identical. The optimisation is a generalisation of the convolutional density estimation scheme presented in [18]. The likelihood may be expressed as[17]

$$\begin{aligned} p(\mathbf{o}_t|\mathbf{q}_t) &= \mathcal{N}\left(\mathbf{o}_t; \sum_{s=1}^{S} \boldsymbol{\mu}^{(q_t^{(s)})}, \boldsymbol{\Sigma}^{(q_t^{(S)})}\right) \\ &= \mathcal{N}\left(\mathbf{o}_t; \mathbf{M}^{(\mathbf{q}_t)}\boldsymbol{\lambda}, \boldsymbol{\Sigma}^{(q_t^{(S)})}\right) \end{aligned} \tag{49}$$

where

$$\mathbf{M}^{(\mathbf{q}_t)} = \left[\begin{array}{ccc} \boldsymbol{\mu}^{(q_t^{(1)})} & \ldots & \boldsymbol{\mu}^{(q_t^{(S)})} \end{array}\right] \tag{50}$$

$$\boldsymbol{\lambda} = \left[\begin{array}{ccc} 1 & \ldots & 1 \end{array}\right]^T \tag{51}$$

Following the scheme described in [18] the scheme is optimised using EM. Differentiating with respect to mean c of stream s ($s \neq S$), $\boldsymbol{\mu}^{(s_c)}$, yields

$$\frac{\partial \mathcal{Q}(\hat{\mathcal{M}}, \mathcal{M})}{\partial \boldsymbol{\mu}^{(c)}} = \sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(s)}=c\right\}} \gamma_t^{(\mathbf{q})} \boldsymbol{\Sigma}^{(q^{(S)})-1} \left(\mathbf{o}_t - \mathbf{M}^{(\mathbf{q})}\boldsymbol{\lambda}\right) \tag{52}$$

where $\hat{\mathcal{M}}$ is the "original" model parameters used to obtain the set of posteriors. When $s = S$

$$\frac{\partial \mathcal{Q}(\hat{\mathcal{M}}, \mathcal{M})}{\partial \boldsymbol{\mu}^{(c)}} = \sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(S)}=c\right\}} \gamma_t^{(\mathbf{q})} \boldsymbol{\Sigma}^{(c)-1} \left(\mathbf{o}_t - \mathbf{M}^{(\mathbf{q})}\boldsymbol{\lambda}\right) \tag{53}$$

There are now a set of linear equations in the complete meta-space of the factorial HMM. For complex systems this becomes impractical. In practice optimisation schemes similar to the more complex transforms may be appropriate. It is easy to show that for $s \neq S$

$$\boldsymbol{\mu}^{(c)} = \left(\sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(s)}=c\right\}} \gamma_t^{(\mathbf{q})} \boldsymbol{\Sigma}^{(q^{(S)})-1}\right)^{-1} \left(\sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(s)}=c\right\}} \gamma_t^{(\mathbf{q})} \boldsymbol{\Sigma}^{(q^{(S)})-1} \left(\mathbf{o}_t - \sum_{i\neq s} \boldsymbol{\mu}^{(q^{(i)})}\right)\right) \tag{54}$$

and for $s = S$

$$\boldsymbol{\mu}^{(c)} = \frac{\sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(S)}=c\right\}} \gamma_t^{(\mathbf{q})} \left(\mathbf{o}_t - \sum_{i\neq S} \boldsymbol{\mu}^{(q^{(i)})}\right)}{\sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(S)}=c\right\}} \gamma_t^{(\mathbf{q})}} \tag{55}$$

The covariance matrix may be estimated as

$$\boldsymbol{\Sigma}^{(c)} = \frac{\sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(S)}=c\right\}} \gamma_t^{(\mathbf{q})} \left(\mathbf{o}_t - \mathbf{M}^{(\mathbf{q})}\boldsymbol{\lambda}\right)\left(\mathbf{o}_t - \mathbf{M}^{(\mathbf{q})}\boldsymbol{\lambda}\right)^T}{\sum_{t=1}^{T} \sum_{\mathbf{q}:\left\{q_t^{(S)}=c\right\}} \gamma_t^{(\mathbf{q})}} \tag{56}$$

[17] Here for simplicity only a single Gaussian component per state is considered. The generalisation to a GMM per state is trivial.

There is an additional generalisation of the linear stream combination scheme that allows the covariance matrix to be a combination of the stream covariances. Rather than using a simple linear combination, it is simpler to use

$$\boldsymbol{\Sigma}^{(\mathbf{q})-1} = \left(\sum_{s=1}^{S} \mathbf{A}^{(q^{(s)})}\right)\left(\sum_{s=1}^{S} \mathbf{A}^{(q^{(s)})}\right)^{T} \tag{57}$$

where $\mathbf{A}^{(q^{(s)})}$ is the "inverse" of the covariance matrix of state $q^{(s)}$. In common with semi-tied transforms and the filter model transform, this matrix becomes full when estimated in an ML fashion, rather than having the standard symmetric form of a covariance matrix.nThe auxilliary function for this optimisation may be written as

$$\mathcal{Q}(\hat{\mathcal{M}},\mathcal{M}) = K + \sum_{t=1}^{T} \sum_{N^1,\dots,N^{(S)}} \left(\gamma_t^{(\mathbf{q})} \log\left(\left(\sum_{s=1}^{S} \mathbf{a}_i^{q^{(s)}} \mathbf{c}_i^{(\mathbf{q})T}\right)^2\right) - \sum_{j=1}^{d}\sum_{s=1}^{S}\sum_{u=1}^{S} \mathbf{a}_j^{q^{(s)}} \mathbf{G}^{(\mathbf{q})} \mathbf{a}_j^{q^{(u)}T}\right) \tag{58}$$

where

$$\mathbf{G}^{(\mathbf{q})} = \sum_{t=1}^{T} \gamma_t^{(\mathbf{q})} \left(\mathbf{o}_t - \mathbf{M}^{(\mathbf{q})}\boldsymbol{\lambda}\right)\left(\mathbf{o}_t - \mathbf{M}^{(\mathbf{q})}\boldsymbol{\lambda}\right) \tag{59}$$

Differentiating with respect to a particular row i of state r of stream s, $\mathbf{a}_i^{(r)}$,

$$\frac{\partial \mathcal{Q}(\hat{\mathcal{M}},\mathcal{M})}{\partial \mathbf{a}_i^{(r)}} = 2\sum_{t=1}^{T} \sum_{\mathbf{q}:\{q_t^{(s)}=r\}} \left(\frac{\gamma_t^{(\mathbf{q})} \mathbf{c}_i^{(\mathbf{q})T}}{\sum_{s=1}^{S} \mathbf{a}_i^{q^{(s)}} \mathbf{c}_i^{(\mathbf{q})T}} - \mathbf{G}^{(\mathbf{q})} \sum_{s=1}^{S} \mathbf{a}_i^{q^{(s)}T}\right) \tag{60}$$

Equating to zero sets up a series of equations which must be satisfied for all streams. Though the model parameters may be estimated using this equation, to date no such system has been implemented.

An interesting relationship with cluster adaptive training [10] (CAT) can be seen from equation 51. Rather than restricting $\boldsymbol{\lambda}$ to be a vector of ones it may be allowed to vary according to the particular speaker. Hence for speaker $\mathcal{S}$

$$p(\mathbf{o}_t|\mathbf{q}_t,\mathcal{S}) = \mathcal{N}\left(\mathbf{o}_t; \mathbf{M}^{(\mathbf{q}_t)}\boldsymbol{\lambda}^{(\mathcal{S})}, \boldsymbol{\Sigma}^{(q_t^{(S)})}\right) \tag{61}$$

where $\boldsymbol{\lambda}^{(\mathcal{S})}$ is the linear combination weights for speaker $\mathcal{S}$. In [10] ML estimation formulae for these parameters are given. Equation 61 is thus a factorial extension of the CAT systems.

B Multiple Transformation Stream Estimation

The estimation of multiple transformation streams is performed a single transformation stream at a time. Consider transformation stream c ($c < S$)

$$\begin{aligned} p(\mathbf{o}_t|\mathbf{q}_t) &= \left(\prod_{s=1}^{S-1} |\det(\mathbf{A}^{(q_t^{(s)})})|\right) \mathcal{N}\left(\mathbf{o}_t^{(S-1)}; \boldsymbol{\mu}^{(q_t^{(S)})}, \boldsymbol{\Sigma}^{(q_t^{(S)})}\right) \qquad (62) \\ &= \left(\prod_{s=1}^{c-1} |\det(\mathbf{A}^{(q_t^{(s)})})|\right) \left(\prod_{s=c+1}^{S-1} |\det(\mathbf{A}^{(q_t^{(s)})})|\right) |\det(\mathbf{A}^{(q_t^{(c)})})| \mathcal{N}\left(\mathbf{o}_t^{(S-1)}; \boldsymbol{\mu}^{(q_t^{(S)})}, \boldsymbol{\Sigma}^{(q_t^{(S)})}\right) \end{aligned}$$

where the following recursion is used to obtain the transformed observation

$$\mathbf{o}_t^{(s)} = \mathbf{W}^{(q_t^{(s)})} \begin{bmatrix} \mathbf{o}_t^{(s-1)} \\ 1 \end{bmatrix} = \mathbf{W}^{(q_t^{(s)})} \boldsymbol{\zeta}_t^{(s-1)} \qquad (63)$$

where $\mathbf{o}_t^{(0)} = \mathbf{o}_t$. Alternatively $\mathbf{o}_t^{(S-1)}$ may be expressed as

$$\begin{aligned} \mathbf{o}_t^{(S-1)} &= \left(\prod_{i=1}^{S-1} \mathbf{A}^{(q_t^{(i)})}\right) \mathbf{o}_t + \sum_{i=1}^{S-1} \left(\left(\prod_{j=i}^{S-2} \mathbf{A}^{(q_t^{(j)})}\right) \mathbf{b}^{(q_t^{(i)})}\right) \\ &= \left(\prod_{i=c+1}^{S-1} \mathbf{A}^{(q_t^{(i)})}\right) \left(\mathbf{A}^{(q_t^{(c)})} \mathbf{o}_t^{(c-1)} + \mathbf{b}^{(q_t^{(c)})}\right) + \sum_{i=c+1}^{S-1} \left(\left(\prod_{j=i}^{S-2} \mathbf{A}^{(q_t^{(j)})}\right) \mathbf{b}^{(q_t^{(i)})}\right) \\ &= \tilde{\mathbf{A}}^{(c)} \left(\mathbf{A}^{(q_t^{(c)})} \mathbf{o}_t^{(c-1)} + \mathbf{b}^{(q_t^{(c)})}\right) + \tilde{\mathbf{b}}^{(c)} \qquad (64) \end{aligned}$$

where

$$\begin{aligned} \tilde{\mathbf{A}}^{(c)} &= \prod_{i=c+1}^{S-1} \mathbf{A}^{(q_t^{(i)})} \qquad (65) \\ \tilde{\mathbf{b}}^{(c)} &= \sum_{i=c+1}^{S-1} \left(\left(\prod_{j=i}^{S-2} \mathbf{A}^{(q_t^{(j)})}\right) \mathbf{b}^{(q_t^{(i)})}\right) \qquad (66) \end{aligned}$$

Finally the likelihood may be expressed as

$$p(\mathbf{o}_t|\mathbf{q}_t) = \tilde{d}^{(c)} |\det(\mathbf{A}^{(q_t^{(c)})})| \mathcal{N}\left(\tilde{\mathbf{A}}^{(c)} \left(\mathbf{A}^{(q_t^{(c)})} \mathbf{o}_t^{(c-1)} + \mathbf{b}^{(q_t^{(c)})}\right) + \tilde{\mathbf{b}}^{(c)}; \boldsymbol{\mu}^{(q_t^{(S)})}, \boldsymbol{\Sigma}^{(q_t^{(S)})}\right) \qquad (67)$$

where

$$\tilde{d}^{(c)} = \left(\prod_{s=1}^{c-1} |\det(\mathbf{A}^{(q_t^{(s)})})|\right) \left(\prod_{s=c+1}^{S-1} |\det(\mathbf{A}^{(q_t^{(s)})})|\right) \qquad (68)$$

This expression has the same form constrained MLLR transform optimisation using a full-covariance matrix system. This is a trivial modification to the re-estimation formulae given in [7]. The update formulae in [7] make use of the standard forward-backward algorithm for HMM. The extension of this algorithm to multiple streams is described in [13]. For highly complex multiple stream transformation stream systems the approximate schemes also described in [13] may be used.

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] E. Bocchieri. Vector quantization for efficient computation of continuous density likelihoods. In *Proc. ICASSP*, volume II, pages II–692–II–695, Minneapolis, 1993.

[3] H Bourlard and S Dupont. A new ASR approach based on independent processing and combination of partial frequency bands. In *Proceedings ICSLP*, pages 422–425, 1996.

[4] S B Davis and P Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions ASSP*, 28:357–366, 1980.

[5] 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.

[6] V D Diakoloukas and V V Digalakis. Maximum-likelihood stochastic-transformation adaptation of hidden Markov models. *IEEE Transactions Speech and Audio Processing*, 7:177–187, 1999.

[7] M J F Gales. Adapting semi-tied full-covariance matrix HMMs. Technical Report CUED/F-INFENG/TR298, Cambridge University, 1997. Available via anonymous ftp from: svr-ftp.eng.cam.ac.uk.

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

[9] M J F Gales. Semi-tied covariance matrices for hidden Markov models. *IEEE Transactions Speech and Audio Processing*, 7:272–281, 1999.

[10] M J F Gales. Cluster adaptive training of hidden Markov models. *IEEE Transactions Speech and Audio Processing*, 8:417–428, 2000.

[11] M J F Gales and S J Young. Robust speech recognition using parallel model combination. *IEEE Transactions Speech and Audio Processing*, 4:352–359, 1996.

[12] Z Ghahramani and G Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report Tech. Rep. CRG-TR96-1, University of Toronto, Canada, 1997.

[13] Z Ghahramani and M I Jordan. Factorial hidden Markov models. *Machine Learning*, 29:245–275, 1997.

[14] R A Gopinath, B Ramabhadran, and S Dharanipragada. Factor analysis invariant to linear transformations of data. In *Proceedings ICSLP*, pages 397–400, 1998.

[15] T Hain, P C Woodland, G Evermann, and D Povey. The CU-HTK March 2000 HUB5E transcription system. In *Proceedings of the Speech Transcription Workshop*, 2000.

[16] 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.

[17] B T Logan and P J Moreno. Factorial HMMs for acoustic modeling. In *Proceedings ICASSP*, pages 813–816, 1998.

[18] S Matsoukas and G Zavaliagkos. Convolutional density estimation in hidden Markov models for speech recognition. In *Proc. ICASSP*, 1999.

[19] N Mirghafori. *A Multi-Band Approach to Automatic Speech Recognition*. PhD thesis, ICSI, UC Berkely, 1999.

[20] L R Neumeyer, A Sankar, and V V Digalakis. A comparative study of speaker adaptation techniques. In *Proceedings Eurospeech*, pages 1127–1130, 1995.

[21] H Nock. *Techniques for modelling Phonological Processes in Automatic Speech Recognition.* PhD thesis, Cambridge University, 2001.

[22] H J Nock and S J Young. Loosely coupled HMMs for ASR. In *Proceedings ICSLP*, 2000.

[23] J J Odell, V Valtchev, P C Woodland, and S J Young. A one pass decoder design for large vocabulary recognition. In *Proceedings ARPA Workshop on Human Language Technology*, pages 405–410, 1994.

[24] J Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[25] D B Rubin and D T Thayer. EM algorithms for ML factor analysis. *Psychometrika*, 47:69–76, 1982.

[26] L K Saul and M I Jordan. Mixed memory Markov models. *Machine Learning*, 37:75–87, 1999.

[27] A P Varga and R K Moore. Hidden Markov model decomposition of speech and noise. In *Proceedings ICASSP*, pages 845–848, 1990.

[28] P C Woodland, J J Odell, V Valtchev, and S J Young. The development of the 1994 HTK large vocabulary speech recognition system. In *Proceedings ARPA Workshop on Spoken Language Systems Technology*, pages 104–109, 1995.

[29] P C Woodland and D Povey. Very large scale MMIE training for conversational telephone speech recognition. In *Proceeding of the 2000 Speech Transcription Workshop*, June 2000.

[30] S J Young, J Jansen, J Odell, D Ollason, and P Woodland. *The HTK Book (for HTK Version 2.0).* Cambridge University, 1996.

[31] S J Young, J J Odell, and P C Woodland. Tree-based state tying for high accuracy acoustic modelling. In *Proceedings ARPA Workshop on Human Language Technology*, pages 307–312, 1994.

[32] G Zweig. *Speech Recognition with Dynamic Bayesian Networks.* PhD thesis, ICSI, UC Berkely, 1999.

[33] G Zweig and S Russell. Probabilistic modeling with Bayesian networks for ASR. In *Proceedings ICSLP*, pages 858–901, 1998.