# MLLR: A SPEAKER ADAPTATION TECHNIQUE FOR LVCSR

*Jonathan E. Hamaker*

Department for Electrical and Computer Engineering
Mississippi State University, Mississippi State, MS 39762
hamaker@isip.msstate.edu

## ABSTRACT

In typical state-of-the-art large vocabulary conversational speech recognition (LVCSR) systems a single model is developed using data from a large number of speakers to cover the variance across dialects, speaking styles, etc. With this, we expect that our systems will generalize well to any particular speaker. However, from experience we know that there are speakers who are poorly modeled using this paradigm. Thus, it is advantageous to adapt the system, during run-time, to the new speaker. In this paper we discuss methods for accomplishing this goal. Most of the effort is spent in describing one popular method that uses a maximum likelihood linear regression (MLLR) approach to speaker adaptation. A derivation of the relevant reestimation equations is provided as well as a discussion of computational and data sufficiency issues.

## 1. INTRODUCTION

Commercially available dictation systems have recently hit the speech products market. These have, for the most part, received rave reviews from users. Most of these systems claim to work well out of the box but perform better as the user performs more dictation with it. This indicates that the systems used in these applications are somehow adjusting to the speaker — also that the speaker is adjusting to the subtleties of getting the application to work. It is this phenomena that we will discuss in this paper.

Speaker-independent recognition systems have been developed to the point that they perform very well for LVCSR in the general case. However, speaker-independent systems, in general, are known to have poorer performance than systems with speaker-dependent models [1, 2]. The main reason for this is that speaker-independent systems are discarding the knowledge that the same speaker is, in fact, speaking every utterance. In doing so, the system is negating the ability of the models to describe the peculiarities of each specific speaker (vocal tract shape and length, accent, etc.) in favor of a general model of any speaker.

On the other hand, there is a very large problem with developing such a speaker-dependent system: doing so would require a large amount of training data from every speaker involved which is impractical for most applications. There are vast amounts of training data available for speaker-independent tasks such as SWITCHBOARD [3]. This provides clear motivation for techniques which would allow us to adapt the speaker-independent models to a new speaker using a small amount of adaptation data. From this need, there have been many attempts to develop robust speaker adaptation techniques.

## 2. SPEAKER ADAPTATION

The basic idea of speaker adaptation can be seen in Figure 1. Essentially, we want to use a a small amount of adaptation data as possible to change our recognition system such that they model as much of the speaker-specific information as possible [4]. Many approaches have been developed which try to produce this effect.

Speaker adaptation techniques for HMM-based recognition systems fall into two basic categories. The first of these employs methods which transform the input speech of the new speaker to a vector space that is common with the training speech. These are known as **spectral mapping techniques**. Second are methods which transform the model parameters to better match the characteristics of the adaptation data. These techniques are known as **model mapping approaches**. The following sections describe each of these, in brief.

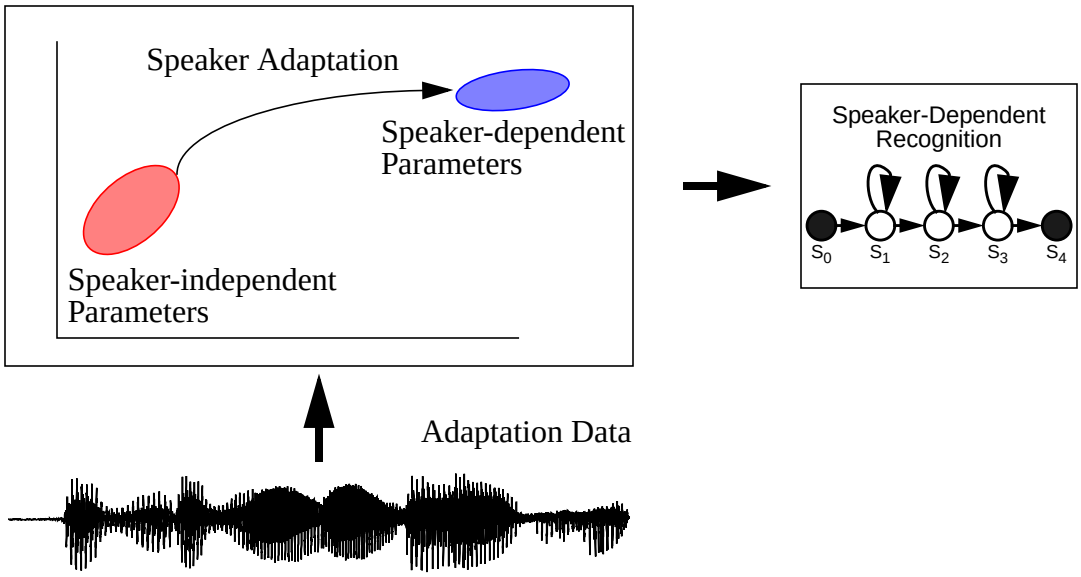



Figure 1. A high-level representation of the speaker adaptation process. The speaker adaptation process uses the adaptation data to affect the modeling process such that the models are a closer match to the adaptation data.

### 2.1. Spectral Mapping Approach

The spectral mapping approach is based on the belief that a recognition system can be improved by matching the new speaker's features vectors to the vectors of the training data [5]. The mapping is designed so that the difference between the reference vector set and the mapped vector set is minimized. These differences are due to the spectral differences of the speakers' speech production systems (e.g. vocal tract length and shape).

Initial attempts at spectral mapping adaptation were used in the spectral template matching systems [6, 7, 8]. These consider the template to be from the reference speaker and automatically generate a transformation to minimize the difference between the new speaker and the reference speaker [5]. Other approaches [9] have mapped both the reference data and the new speaker's data into a common vector set which is said to maximally correlate the two. A variation on these methods which is similar to speaker normalization uses a transform to map each speaker in the speaker-independent training set onto a reference speaker [10, 11]. Thus, the models generated act as speaker-dependent models. This approach is illustrated in Figure 2 and is commonly referred to as a speaker normalization technique.

### 2.2. Model Mapping Approach

The aim of spectral mapping is to improve the match between the reference speaker and new speaker. However, this goal does not explicitly try to increase the accuracy of the models for the new speaker and, thus, does not take full advantage of the adaptation data. This is where the model mapping approach attempts to make its improvements. Rather than trying to map all speakers to one space, the model mapping approach adjusts the model parameters to best represent the new speaker as illustrated in Figure 3.

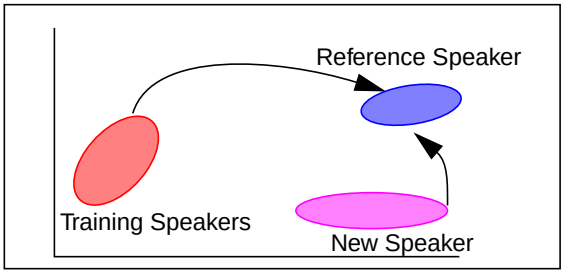



Figure 2. Spectral Mapping approach where both the training speakers and new speaker feature vectors are mapped to a common space which maximizes the correlation between the two.

A common example of the model mapping approach is the bayesian MAP (maximum a posteriori) technique for adapting HMMs. In a MAP approach, the transformation is chosen such that the new model parameters maximize a likelihood function,

$$P(\lambda|O) = \frac{P(O|\lambda)P(\lambda)}{P(O)}, \quad (1)$$

where $O$ is the adaptation observation sequence and $\lambda$ is the parameter set defining the distribution. Different methods have been used to estimate the value of $\lambda$ including a segmental K-means approach [12] and an EM-based approach [13]. Most of these MAP approaches are limited in that they only adapt the parameters that are directly observed in the adaptation data.

A more successful model mapping technique is maximum likelihood linear regression (MLLR) [1, 5, 14]. MLLR was designed to overcome the disadvantages of both the spectral mapping and model mapping techniques. MLLR is a transform-based method which adapts the model parameters like the MAP-based adaptation but, using transform tying, is robust enough to produce effects from a small amount of training data. This approach was developed from work by Hewitt [15] which applied a least squares regression to adapt templates in dynamic time warping. MLLR extends this idea to the continuous density HMMs and uses maximum likelihood (ML) to optimize the regression.

Two issues that must be addressed when discussing model mapping approaches are the training modes (supervised verses unsupervised) and the adaptation mode (incremental verses batch). In a supervised training mode the recognition system is given the correct transcription and has only to align the user's speech to that transcription. In unsupervised adaptation the recognizer feeds itself, perhaps including recognition errors. Obviously, the supervised mode is preferred when available. This is why commercial dictation systems employ an enrollment process where the user recites some pre-transcribed sentences.

The adaptation mode describes when the adaptation takes place and what models are employed to produce the hypotheses used for adaptation. In incremental mode, the models are adapted quite often and the adapted models are used to produce the hypotheses for the next adaptation. This is the typical method seen in real-time systems that use adaptation. Batch mode is similar to a training run where hypotheses for the entire adaptation set are stored and then used to iteratively update the adapted models. Again, this is similar to the enrollment process in commercial systems.

### 2.3. Performance Equals Motivation

Table 1 demonstrates why adaptation techniques have become popular in recent years. These systems

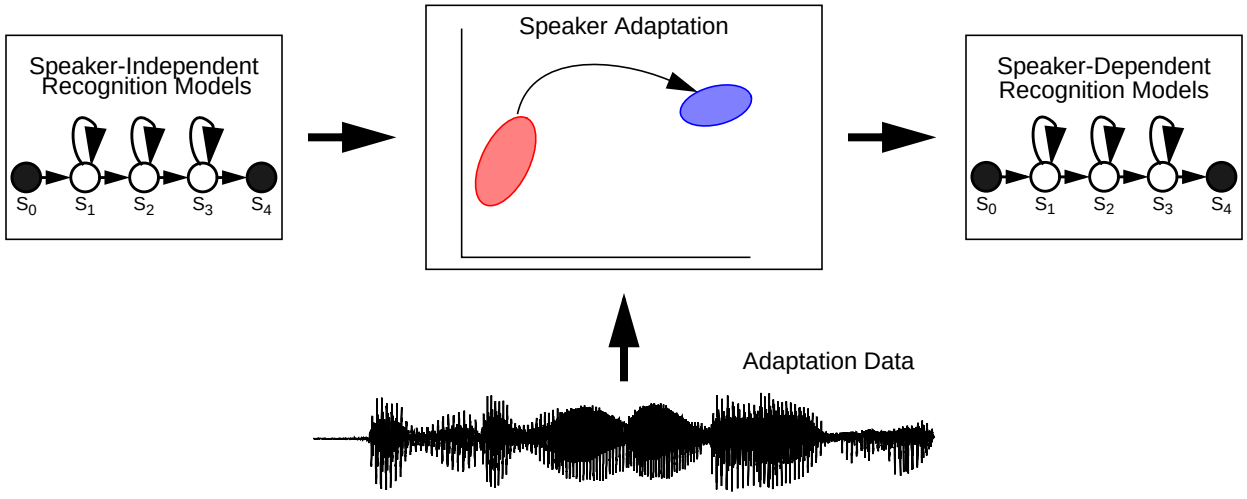



Figure 3. Representation of a model mapping approach where speaker-independent models are mapped to speaker-dependent models.

use combinations of vocal tract length normalization (VTLN — a speaker normalization technique) and MLLR to adapt a medium vocabulary triphone system [18]. On average, the combined normalization and adaptation yielded a 35% reduction in word error rate (WER) for supervised adaptation and a 15% reduction for unsupervised adaptation. Similar, though not quite as good, results hold for large vocabulary tasks such as SWITCHBOARD and Broadcast News.

## 3. MLLR BASICS

MLLR produces a set of regression-based transforms that are used to tune the HMM emission parameters using some given adaptation data. MLLR is able to build robust adaptation transforms even for models that are not seen in the adaptation data using transform sharing. This helps to eliminate the problem of limited adaptation data. With very little data, a single global transform can be used for all models. As more data becomes available, more fine-grain transforms can be applied. In Figure 4 we see the basic approach of MLLR which is to take a speaker-independent model (the solid red ovals) and use a transform to move the model space toward a speaker-dependent model (the striped blue ovals). Commonly, only a mean adaptation is performed since it is assumed that the primary difference between speakers is in the average position of phones in the acoustic space [14]. This is the same reasoning given in many VQ/HMM adaptation schemes [16, 17]. A covariance adaptation is less commonly used and its effects are less profound than the mean adaptation [19].

### 3.1. Maximum Likelihood Estimation

In HMM-based recognition systems, we need to estimate the parameters of the system so that they yield the best performance possible. Ideally this would be done so as to minimize the error rate. However, there is usually no closed-form description of this problem so traditional optimization techniques (gradient descent, for example) will not work. A more common approach is to modify the system so that the probability of the training data given the new model is maximized. For computational convenience the probability is often described as a likelihood function. So, we arrive at estimating the parameters of the model to maximize a likelihood function.

We typically solve an MLE problem using the Expectation Maximization (EM) algorithm [20]. EM determines the estimated parameters of a model set such that the newly estimated parameters are guaranteed to increase the value of some specified function. This is described as:

$$f_{mle}(\hat{\lambda}) \geq f_{mle}(\lambda) \ , \tag{2}$$

where $\lambda$ are the parameters of the model. Rather than trying to maximize this function directly, we often formulate an auxiliary function that is more computationally tractable and which has nice convergence properties. For speech training systems, this takes the form:

$$Q(\lambda, \hat{\lambda}) = \sum_{\underline{\theta} \in \underline{\Theta}} P(\underline{O}, \underline{\theta}|\lambda) \log\{P(\underline{O}, \underline{\theta}|\hat{\lambda})\} \tag{3}$$

where $\underline{\Theta}$ contains all possible state sequences leading to the recognition of $\underline{O}$. Note that (3) amounts to a cross-entropy function where the convexity of the entropy function is exploited to guarantee maximization. This form should be familiar to those who have studied Baum-Welch training [20]. Baum proved that this form was guaranteed to increase the probability function.

For HMMs, the probabilities are related to both the transition probabilities and the state emission probabilities. Thus, we can expand the auxiliary function as

$$\sum_{\underline{\theta} \in \underline{\Theta}} L(\underline{O}, \underline{\theta}|\lambda) \left[ \begin{array}{c} \sum_{t=1}^{T} \log(\hat{\text{transition}} \text{ prob.}) + \\ \sum_{t=1}^{T} \log \hat{b}_{\theta_t}(\underline{o}_t) \end{array} \right] \tag{4}$$

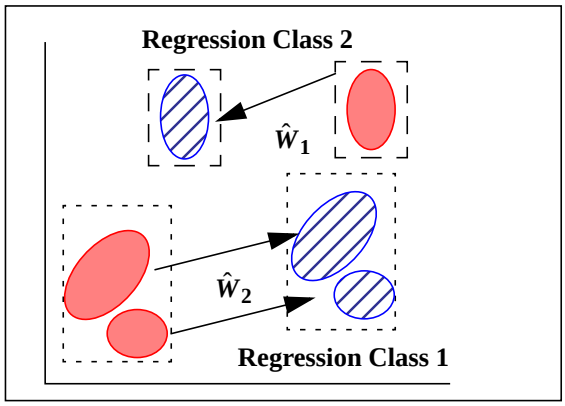
This auxiliary function is differentiated with respect to each parameter of interest and set to zero to determine a closed-form solution for the parameter reestimation.

### 3.2. Linear Regression

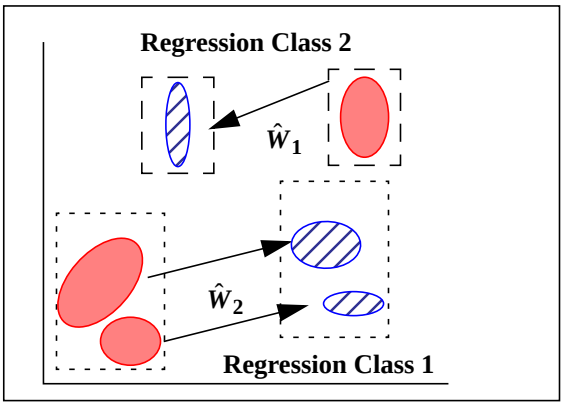
In linear regression, we desire to explain a set of $n$ output variables, $(y_1, y_2, \ldots, y_n)$ as a linear

| Push-to-talk Data with ~4600 word Vocabulary | | | | |
|---|---|---|---|---|
| **Speaker** | **SI** | **MLLR Sup/UnSup** | **VTLN Sup/UnSup** | **MLLR/VTLN Sup/UnSup** |
| **Meba** | **10.4%** | **4.7 / 7.3%** | **10.4 / 8.6%** | **5.6 / 6.9%** |
| **Mfmm** | **20.5%** | **16.7 / 20.5%** | **19.3 / 21.6%** | **13.4 / 20.1%** |
| **Mofc** | **11.8%** | **8.0 / 11.8%** | **9.4 / 8.5%** | **5.2 / 8.5%** |
| **Macc** | **27.1%** | **22.5 / 27.7%** | **26.5 / 26.1%** | **21.3 / 25.9%** |
| **Mrnn** | **31.5%** | **18.8 / 30.2%** | **26.5 / 28.7%** | **18.2 / 28.5%** |
| **Fcba** | **14.0%** | **12.1 / 16.7%** | **16.7 / 14.4%** | **10.7 / 13.9%** |
| **Fnba** | **15.5%** | **10.4 / 14.9%** | **12.3 / 13.3%** | **10.4 / 13.3%** |
| **Fmcs** | **25.0%** | **16.4 / 23.1%** | **21.6 / 22.1%** | **16.0 / 21.4%** |
| **Fmgl** | **25.0%** | **20.4 / 27.4%** | **22.4 / 22.5%** | **13.2 / 22.5%** |
| **Avg** | **21.8%** | **15.3 / 21.3%** | **19.1 / 19.4%** | **14.0 / 18.6%** |

Table 1. Effects of speaker adaptation and speaker normalization on medium vocabulary recognition.




**Mean Adaptation**




**Mean and Variance Adaptation**

Figure 4. Adapting speaker-independent models to speaker-dependent models using an MLLR approach. Note that mean adaptation shifts the location of the models means in the space, while (co)variance adaptation changes the shape of the distributions.

combination of m explanatory variables, $(x_1, x_2, ..., x_m)$. We can define this set of simultaneous equations as

$$\begin{aligned} y_1 &= a_{10} + a_{11}x_1 + \ldots + a_{1m}x_m \\ y_2 &= a_{20} + a_{21}x_1 + \ldots + a_{2m}x_m \\ &\ldots \\ y_n &= a_{n0} + a_{n1}x_1 + \ldots + a_{nm}x_m \end{aligned} \quad . \tag{5}$$

In matrix form this is

$$\begin{bmatrix} y_1 \\ y_2 \\ \ldots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{10} & a_{11} & \ldots & a_{1m} \\ a_{20} & a_{21} & \ldots & a_{2m} \\ \ldots & \ldots & \ldots & \ldots \\ a_{no} & a_{n1} & \ldots & a_{nm} \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \ldots \\ x_m \end{bmatrix} \tag{6}$$

or $\underline{y} = \underline{\underline{W}}\underline{x}$.

### 3.3. Transform Sharing

Ideally, we would use one adaptation transform per Gaussian model in an HMM system so that all of the differences between the speaker-independent model and the speaker-dependent model could be accurately found. However, in practice, this would require too much adaptation data to accurately estimate the adapted models. For this reason, a form of transform sharing is usually employed where sets of Gaussians are pooled together and are adapted by a single transform. In this scheme, components with little or no adaptation data can be adapted with the data pooled from similar models.

A common approach for transform pooling is to use a binary regression tree as shown in Figure 5. A pooling algorithm is needed to determine which components are pooled to each node. A centroid splitting algorithm is often used which attempts to split the components at a node into two sets of components located at the two children of that node. This is done in a way that ensures the components that are closest together in the probability space are pooled to the same child node [19].

## 4. TRANSFORM ESTIMATION

Use of the EM algorithm typically involves an iterative process whereby the parameters of the system are iteratively redefined to maximize the likelihood function. It is desirable to have closed-form reestimation formulae to generate the new parameters on every iteration. In this section we derive those formulae for the mean transform and the covariance transform in the MLLR scheme. These follow from [14] and [19] respectively.

### 4.1. Mean Transform Estimation

Let the mean for a single mixture component be an n-dimensional vector defined as $\underline{\mu}_s$, Then, we define the adapted mean estimate as

$$\hat{\underline{\mu}}_s = \underline{\underline{W}}_s \underline{\xi}_s \tag{7}$$

where $\underline{\underline{W}}_s$ is an n x (n+1) transformation matrix and $\underline{\xi}_s = [\overline{w}, \underline{\mu}_{s1}, ..., \underline{\mu}_{sn}]^t$ is the extended mean vector. $\overline{w}$ is the offset indicator so that $w = 1$ indicates an offset and $w = 0$ indicates no offset. For Gaussian probability models, this gives an adapted mixture density function of

$$b_s(\underline{o}) = \frac{1}{(2\pi)^{\frac{n}{2}} \left|\underline{\underline{\Sigma}}_s\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(\underline{o} - \underline{\underline{W}}_s\underline{\xi}_s)^t \, \underline{\underline{\Sigma}}_s^{-1} (\underline{o} - \underline{\underline{W}}_s\underline{\xi}_s)} \tag{8}$$

A maximum likelihood estimate of each $\underline{\underline{W}}_s$ matrices is made, maximizing the likelihood of the adapted model set generating the adaptation data.

#### 4.1.1. General form

Assume the adaptation data is a series of T observations, $\underline{O} = o_1, ..., o_T$. Since we are interested in reestimating only the transformation matrix, we can ignore the terms in the auxiliary function (4) due to the transition probabilities. This gives an auxiliary function of the form

$$\text{constant} + \sum_{\underline{\theta} \in \underline{\Theta}} \sum_{t=1}^{T} P(\underline{O}, \underline{\theta}|\lambda) \log \hat{b}_{\theta_t}(\underline{o}_t) . \tag{9}$$

We can define the posterior probability of occupying state s at time t given that the observation sequence $\underline{O}$ is generated as

$$\gamma_s(t) = \frac{1}{P(\underline{O}|\lambda)} \sum_{\underline{\theta} \in \underline{\Theta}} P(\underline{O}, \theta_t = s|\lambda) . \tag{10}$$

This is more commonly known as the state occupancy probability. Let S be the set of all states in the system. Then we can sum the marginal

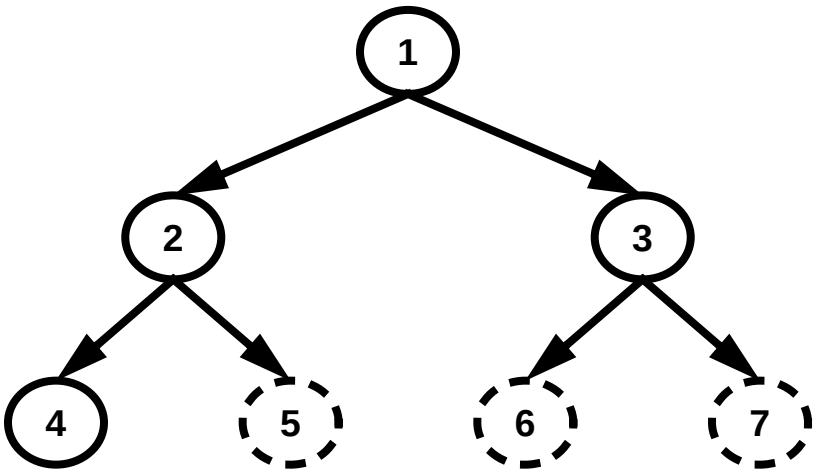



Figure 5. A binary regression tree for pooling components to be adapted. The leaf nodes (4, 5, 6, 7) are the base regression classes. The dashed circles represent nodes that have insufficient training data and, are thus, pooled to the next higher level in the tree. The solid circles have sufficient data for estimating a transform.

probabilities across the set of states to get the total probability. Thus the auxiliary function expands to

$$\text{constant} + P(\underline{O}|\lambda) \sum_{j=1}^{S} \sum_{t=1}^{T} \gamma_j(t) \log \hat{b}_j(\underline{o}_t) . \quad (11)$$

We now differentiate $Q(\lambda, \hat{\lambda})$ with respect to $\underline{\underline{\hat{W}}}_s$ and obtain the differential

$$P(\underline{O}|\lambda) \frac{d}{d\underline{\underline{\hat{W}}}_s} \sum_{j=1}^{S} \sum_{t=1}^{T} \gamma_j(t) \log \hat{b}_j(\underline{o}_t) . \quad (12)$$

expanding $\hat{b}_j(\underline{o}_t)$ in (12) as a Gaussian we have

$$-\frac{1}{2} P(\underline{O}|\lambda) \frac{d}{d\underline{\underline{\hat{W}}}_s} \sum_{j=1}^{S} \sum_{t=1}^{T} \gamma_j(t) \left[ \begin{matrix} n \log(2\pi) + \\ h(\underline{o}_t, j) + \log|\underline{\underline{\Sigma}}_j| \end{matrix} \right] \quad (13)$$

where

$$h(\underline{o}_t, j) = (\underline{o}_t - \underline{\underline{\hat{W}}}_j \underline{\xi}_j)^t \underline{\underline{\Sigma}}_j^{-1} (\underline{o}_t - \underline{\underline{\hat{W}}}_j \underline{\xi}_j) . \quad (14)$$

Since $h(\underline{o}_t, s)$ is the only term in the summation dependent on $\underline{\underline{W}}_s$, the differential of the auxiliary function (13) reduces to

$$-\frac{1}{2} P(\underline{O}|\lambda) \sum_{t=1}^{T} \gamma_s(t) \frac{d}{d\underline{\underline{\hat{W}}}_s} h(\underline{o}_t, s) \quad (15)$$

or

$$P(\underline{O}|\lambda) \sum_{t=1}^{T} \gamma_s(t) \underline{\underline{\Sigma}}_s^{-1} (\underline{o}_t - \underline{\underline{\hat{W}}}_s \underline{\xi}_s) \underline{\xi}_s^t . \quad (16)$$

To optimize this function we set (16) to zero and group terms so that we have known quantities on one side of the equation and unknown quantities (i.e. terms multiplied by $\underline{\underline{\hat{W}}}_s$) on the other. This results in

$$\sum_{t=1}^{T} \gamma_s(t) \underline{\underline{\Sigma}}_s^{-1} \underline{o}_t \underline{\xi}_s^t = \sum_{t=1}^{T} \gamma_s(t) \underline{\underline{\Sigma}}_s^{-1} \underline{\underline{\hat{W}}}_s \underline{\xi}_s \underline{\xi}_s^t . \quad (17)$$

(17) is the general form for computing $\underline{\underline{\hat{W}}}_s$. The solution can only be found for specific types of problems - most notably for diagonal covariance matrices.

### 4.1.2. Closed-form solution

We now derive the closed-form solution for the mean transform when all covariance matrices are diagonal. If $\underline{\underline{\hat{W}}}_s$ is shared by R states, $\{s_1, s_2, ..., s_R\}$, then the general form in (17) expands to

$$\sum_{t=1}^{T} \sum_{r=1}^{R} \gamma_{s_r}(t) \underline{\underline{\Sigma}}_{s_r}^{-1} \underline{o}_t \underline{\xi}_{s_r}^t = \sum_{t=1}^{T} \sum_{r=1}^{R} \gamma_{s_r}(t) \underline{\underline{\Sigma}}_{s_r}^{-1} \underline{\underline{\hat{W}}}_s \underline{\xi}_{s_r} \underline{\xi}_{s_r}^t . \quad (18)$$

We rewrite this as

$$\sum_{t=1}^{T}\sum_{r=1}^{R}\gamma_{s_r}(t)\underline{\Sigma}_{s_r}^{-1}\underline{o}_t\underline{\xi}_{s_r}^{t} = \sum_{r=1}^{R}\underline{\underline{V}}^{(r)}\underline{\underline{\hat{W}}}_s\underline{\underline{D}}^{(r)} \quad (19)$$

where $\underline{\underline{V}}^{(r)}$ is the state distribution inverse covariance matrix scaled by the state occupancy probability and $\underline{\underline{D}}^{(r)}$ is the outer product of the extended mean vector

$$\underline{\underline{V}}^{(r)} = \sum_{t=1}^{T}\gamma_{s_r}(t)\underline{\Sigma}_{s_r}^{-1} \quad (20)$$

$$\underline{\underline{D}}^{(r)} = \underline{\xi}_{s_r}\underline{\xi}_{s_r}^{t}. \quad (21)$$

Note that (21) defines a singular matrix.

Let the right hand side of (19) be a n x (n+1) matrix, $\underline{\underline{Y}}$ and let the elements of $\underline{\underline{Y}}$, $\underline{\underline{V}}^{(r)}$, $\underline{\underline{W}}_s$, $\underline{\underline{D}}^{(r)}$ be $y_{ij}$, $v_{ij}^{(r)}$, $w_{ij}$, and $d_{ij}^{(r)}$ respectively. Then we can write

$$y_{ij} = \sum_{p=1}^{n}\sum_{q=1}^{n+1}w_{pq}\sum_{r=1}^{R}v_{ip}^{(r)}d_{qj}^{(r)}. \quad (22)$$

Since $\underline{\underline{D}}^{(r)}$ is symmetric and since we have specified that all covariances are diagonal then

$$\sum_{r=1}^{R}v_{ip}^{(r)}d_{qj}^{(r)} = \begin{bmatrix}\sum_{r=1}^{R}v_{ii}^{(r)}d_{jq}^{(r)}, \text{ when } i = p\\ 0, \text{ when } i \neq p\end{bmatrix} \quad (23)$$

and

$$y_{ij} = \sum_{q=1}^{n+1}w_{iq}\sum_{r=1}^{R}v_{ii}^{(r)}d_{jq}^{(r)}. \quad (24)$$

Setting

$$g_{jq}^{(i)} = \sum_{r=1}^{R}v_{ii}^{(r)}d_{jq}^{(r)} \quad (25)$$

gives

$$y_{ij} = \sum_{q=1}^{n+1}w_{iq}g_{jq}^{(i)} \quad (26)$$

where $g_{jk}^{(i)}$ are the elements of an (n+1) x (n+1) matrix, $\underline{\underline{G}}^{(i)}$. Note that since $D^{(r)}$ is singular, $\underline{\underline{G}}^{(i)}$ is also singular.

Setting the left hand side of (19) to $\underline{\underline{Z}}$ gives $\underline{\underline{Z}} = \underline{\underline{Y}}$ and

$$z_{ij} = y_{ij} = \sum_{q=1}^{n+1}w_{iq}g_{jq}^{(i)} \quad (27)$$

where $z_{ij}$ and $g_{jq}^{(i)}$ can be computed from the observation vectors and the model parameters. This gives a set of linear reestimation equations

$$\underline{w}_i^{t} = (\underline{\underline{G}}^{(i)})^{-1}\underline{z}_i^{t} \quad (28)$$

where $w_i$ and $z_i$ are the $i^{th}$ rows of $\underline{\underline{\hat{W}}}_s$ and $\underline{\underline{Z}}$ respectively.

#### 4.1.3. Optimizations

Note that the solution to each row involves an extremely expensive computational structure including inversion of singular matrices and numerous matrix multiplies. To reduce this load, diagonal or block-diagonal forms are often assumed for the transform matrix. A diagonal transform is specified as

$$\underline{\underline{\hat{W}}}_s = \begin{bmatrix} w_{1,1} & w_{1,2} & 0 & \dots & 0\\ w_{2,1} & 0 & w_{2,3} & \dots & 0\\ \dots & \dots & \dots & \dots & \dots\\ w_{n,1} & 0 & \dots & 0 & w_{n,n+1}\end{bmatrix} \quad (29)$$

so that each component of the mean undergoes a shift and scaling

$$\hat{\mu}_i = ww_{i,1} + w_{i,i+1}\mu_i. \quad (30)$$

Block-diagonal transforms assume independence amongst subsets of the mean components. For typical LVCSR systems, we may define the transform as

$$\underline{\underline{\hat{W}}}_s = \left[\begin{bmatrix} w_{1,1}\\ \dots\\ w_{n,1}\end{bmatrix}, \begin{bmatrix} A_s & 0 & 0\\ 0 & A_\Delta & 0\\ 0 & 0 & A_{\Delta^2}\end{bmatrix}\right]. \quad (31)$$

Here $A_s$ indicates the basic spectral features, $A_\Delta$ the derivative features, and $A_{\Delta^2}$ the acceleration features.

### 4.2. Covariance Transform Estimation

We define the adapted variance as

$$\underline{\underline{\hat{\Sigma}}}_s = \underline{\underline{B}}_s^{t}\underline{\underline{\hat{H}}}_s\underline{\underline{B}}_s \quad (32)$$

where $\hat{H}_s$ is the transform to be estimated and $B_s$ is the inverse of the Cholesky factor of $\Sigma_s^{-1}$. So,

$$\Sigma_s^{-1} = C_s C_s^t \tag{33}$$

and

$$B_s = C_s^{-1}. \tag{34}$$

Cholesky decomposition is used as it insures that the resulting matrix is non-singular. We, again, use the auxiliary function from (11)

$$\text{constant} + P(O|\lambda) \sum_{j=1}^{S} \sum_{t=1}^{T} \gamma_j(t) \log \hat{b}_j(o_t). \tag{35}$$

Expanding $\log \hat{b}_j(o_t)$ using as (8) and (32), we obtain

$$\log \hat{b}_j(o_t) = -\frac{1}{2}[n\log(2\pi) + \log|\Sigma_j|] + -\frac{1}{2}\log|\hat{H}_j| + -\frac{1}{2}[(o_t - \hat{\mu}_j)^t B_j^{-1} \hat{H}_j^{-1} (B_j^t)^{-1} (o_t - \hat{\mu}_j)] \tag{36}$$

Since $B_j = C_j^{-1}$,

$$\log \hat{b}_j(o_t) = -\frac{1}{2}[n\log(2\pi) + \log|\Sigma_j|] + -\frac{1}{2}\log|\hat{H}_j| + -\frac{1}{2}[(o_t - \hat{\mu}_j)^t C_j \hat{H}_j^{-1} C_j^t (o_t - \hat{\mu}_j)] \tag{37}$$

or

$$\log \hat{b}_j(o_t) = -\frac{1}{2}[n\log(2\pi) + \log|\Sigma_j|] + -\frac{1}{2}\log|\hat{H}_j| + -\frac{1}{2}[(C_j^t o_t - C_j^t \hat{\mu}_j)^t \hat{H}_j^{-1} (C_j^t o_t - C_j^t \hat{\mu}_j)] \tag{38}$$

We then differentiate the auxiliary function, $Q(\lambda, \hat{\lambda})$, with respect to $\hat{H}_s$, set the derivative to zero and group like terms to yield

$$\hat{H}_s = \frac{C_s^t \sum_{t=1}^{T} \gamma_s(t)[(o_t - \hat{\mu}_s)(o_t - \hat{\mu}_s)^t] C_s}{\sum_{t=1}^{T} \gamma_s(t)}. \tag{39}$$

If $\hat{H}_s$ is shared by R states, $\{s_1, s_2, \ldots, s_R\}$ then

$$\hat{H}_s = \frac{\sum_{r=1}^{R} C_{s_r}^t \sum_{t=1}^{T} \gamma_{s_r}(t) \left[ \left( o_t - \hat{\mu}_{s_r} \right)^{\sim} \right] C_{s_r}}{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{s_r}(t)} \tag{40}$$

where $\tilde{x} = xx^t$.

The transformation of the covariance using the estimate for $\hat{H}_s$ results in a full covariance matrix, but the off-diagonal terms in $\hat{H}_s$ can be set to zero and an increase in likelihood is still guaranteed.

## 5. A SIMPLE EXAMPLE

While the preceding derivations are fairly straightforward, it is instructive to see a numerical example that demonstrates the formulae in action. What follows is the computation of a mean transform for a single state recognition system using a two-dimensional acoustic space and diagonal covariances.

Assume that the following defines a single state in a recognition system using a two-dimensional acoustic space with diagonal covariances:

$$\mu_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 4 & 0 \\ 0 & 9 \end{bmatrix}. \tag{41}$$

Now let's say that we have two frames of adaptation data (generally this is not sufficient to generate a robust estimate, but we continue for the sake of example),

$$o_1 = \begin{bmatrix} 4 \\ 3.5 \end{bmatrix}, o_2 = \begin{bmatrix} 4.2 \\ 3.3 \end{bmatrix}. \tag{42}$$

Computing the mean and covariance of the observed data we have

$$\mu_o = \begin{bmatrix} 4.1 \\ 3.4 \end{bmatrix}, \Sigma_o = \begin{bmatrix} 0.02 & -0.02 \\ -0.02 & 0.02 \end{bmatrix}. \tag{43}$$

Recall that, for diagonal covariances, we want to solve the set of functions

$$w_i^t = (G^{(i)})^{-1} z_i^t \tag{44}$$

where $w_i$ and $z_i$ are the $i^{th}$ rows of $\hat{W}_s$ and $Z$

respectively and

$$\underline{Z} = \sum_{t=1}^{T}\sum_{r=1}^{R} \gamma_{s_r}(t)\underline{\Sigma}_{s_r}^{-1}\underline{o}_t\underline{\xi}_{s_r}^{t}. \quad (45)$$

For the sake of example we will define

$$\gamma_1(1) = 0.3, \gamma_1(2) = 0.8. \quad (46)$$

Solving for $\underline{Z}$ in (45) gives

$$\underline{Z} = 0.3\begin{bmatrix}0.25 & 0\\ 0 & 0.111\end{bmatrix}\begin{bmatrix}4\\3.5\end{bmatrix}\begin{bmatrix}1 & 2 & 3\end{bmatrix} + 0.8\begin{bmatrix}0.25 & 0\\ 0 & 0.111\end{bmatrix}\begin{bmatrix}4.2\\3.3\end{bmatrix}\begin{bmatrix}1 & 2 & 3\end{bmatrix} \quad (47)$$

or

$$\underline{Z} = \begin{bmatrix}1.14 & 2.28 & 3.42\\ 0.4096 & 0.8192 & 1.2288\end{bmatrix}. \quad (48)$$

For a diagonal covariance, we defined the elements of $\underline{G}^{(i)}$ by

$$g_{jq}^{(i)} = \sum_{r=1}^{R} v_{ii}^{(r)} d_{jq}^{(r)}, \quad (49)$$

where $q$ ranged from $1, \ldots, n+1$. We expand

$$\underline{V}^{(r)} = \sum_{t=1}^{T} \gamma_{s_r}(t)\underline{\Sigma}_{s_r}^{-1} = 0.3\begin{bmatrix}0.25 & 0\\ 0 & 0.111\end{bmatrix} + 0.8\begin{bmatrix}0.25 & 0\\ 0 & 0.111\end{bmatrix} \quad (50)$$

or

$$V^{(r)} = \begin{bmatrix}0.275 & 0\\ 0 & 0.1221\end{bmatrix} \text{ and} \quad (51)$$

$$\underline{D}^{(r)} = \underline{\xi}_{s_r}\underline{\xi}_{s_r}^{t} = \begin{bmatrix}1\\2\\3\end{bmatrix}\begin{bmatrix}1 & 2 & 3\end{bmatrix} = \begin{bmatrix}1 & 2 & 3\\ 2 & 4 & 6\\ 3 & 6 & 9\end{bmatrix} \quad (52)$$

At this point, we have what we need to solve for $\underline{G}^{(i)}$:

$$\underline{G}^{(1)} = 0.275\begin{bmatrix}1 & 2 & 3\\ 2 & 4 & 6\\ 3 & 6 & 9\end{bmatrix} = \begin{bmatrix}0.275 & 0.550 & 0.825\\ 0.550 & 1.100 & 1.650\\ 0.825 & 1.650 & 2.475\end{bmatrix} \quad (53)$$

and

$$\underline{G}^{(2)} = 0.1221\begin{bmatrix}1 & 2 & 3\\ 2 & 4 & 6\\ 3 & 6 & 9\end{bmatrix} = \begin{bmatrix}0.122 & 0.244 & 0.366\\ 0.244 & 0.488 & 0.733\\ 0.366 & 0.733 & 1.099\end{bmatrix}. \quad (54)$$

Note that both of the $\underline{G}^{(i)}$ are singular so we must use a method such as singular value decomposition to find the inverse.

$$(\underline{G}^{(1)})^{-1} = \begin{bmatrix}1.86\times10^{-2} & 3.71\times10^{-2} & 5.57\times10^{-2}\\ 3.71\times10^{-2} & 7.42\times10^{-2} & 1.11\times10^{-1}\\ 5.57\times10^{-2} & 1.11\times10^{-1} & 1.67\times10^{-1}\end{bmatrix} \quad (55)$$

and

$$(\underline{G}^{(2)})^{-1} = \begin{bmatrix}4.18\times10^{-2} & 8.36\times10^{-2} & 1.25\times10^{-1}\\ 8.36\times10^{-2} & 1.67\times10^{-1} & 2.51\times10^{-1}\\ 1.25\times10^{-1} & 2.51\times10^{-1} & 3.76\times10^{-1}\end{bmatrix} \quad (56)$$

Solving for the rows of $\underline{w}_i^{t}$ yields

$$\underline{w}_1^{t} = \begin{bmatrix}1.86\times10^{-2} & 3.71\times10^{-2} & 5.57\times10^{-2}\\ 3.71\times10^{-2} & 7.42\times10^{-2} & 1.11\times10^{-1}\\ 5.57\times10^{-2} & 1.11\times10^{-1} & 1.67\times10^{-1}\end{bmatrix}\begin{bmatrix}1.14\\2.28\\3.42\end{bmatrix},$$

$$\underline{w}_1^{t} = \begin{bmatrix}2.961\times10^{-1}\\5.922\times10^{-1}\\8.883\times10^{-1}\end{bmatrix} \text{ and} \quad (57)$$

$$\underline{w}_2^{t} = \begin{bmatrix}4.18\times10^{-2} & 8.36\times10^{-2} & 1.25\times10^{-1}\\ 8.36\times10^{-2} & 1.67\times10^{-1} & 2.51\times10^{-1}\\ 1.25\times10^{-1} & 2.51\times10^{-1} & 3.76\times10^{-1}\end{bmatrix}\begin{bmatrix}0.4096\\0.8192\\1.2288\end{bmatrix}$$

$$\underline{w}_2^{t} = \begin{bmatrix}2.396\times10^{-1}\\4.792\times10^{-1}\\7.188\times10^{-1}\end{bmatrix}. \quad (58)$$

Thus,

$$\underline{\underline{\hat{W}}}_1 = \begin{bmatrix} 2.961\times10^{-1} & 5.922\times10^{-1} & 8.883\times10^{-1} \\ 2.396\times10^{-1} & 4.792\times10^{-1} & 7.188\times10^{-1} \end{bmatrix}. \quad (59)$$

We can now compute the adapted mean as

$$\underline{\hat{\mu}}_1 = \underline{\underline{\hat{W}}}_1 \underline{\xi}_1$$

$$\underline{\hat{\mu}}_1 = \begin{bmatrix} 2.961\times10^{-1} & 5.922\times10^{-1} & 8.883\times10^{-1} \\ 2.396\times10^{-1} & 4.792\times10^{-1} & 7.188\times10^{-1} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$\underline{\hat{\mu}}_1 = \begin{bmatrix} 4.145 \\ 3.355 \end{bmatrix}. \quad (60)$$

Recall that

$$\underline{\mu}_o = \begin{bmatrix} 4.1 \\ 3.4 \end{bmatrix}. \quad (61)$$

The mean has moved significantly closer to the observed data mean. Note that the state occupancy probabilities are quite high in this sample case which is why the transformed mean jumped toward the observed data mean after only two observations. In general the state occupancy probabilities will be much lower, giving a more gradual adaptation.

## 6. SUMMARY

In this paper we have derived and illustrated the use of MLLR as a speaker adaptation technique. Using MLLR, we are able to provide an extremely flexible scheme for adapting to small quantities of speaker-dependent data. It should be remembered, though, that MLLR is a maximum likelihood estimator. While there is ample evidence that, in general, maximizing likelihood also reduces word error rate (the ultimate goal of speech recognition), there is no guarantee. An interesting area for future research will be hypothesis based optimization where the recognizer hypothesis is specifically accounted for in the optimization scheme.

## 7. REFERENCES


[1] C. J. Leggetter, and P. C. Woodland, "Flexible Speaker Adaptation Using Maximum Likelihood Linear Regression," *Proceedings of the ARPA Spoken Language Technology Workshop, Barton Creek, 1995.*

[2] X. D. Huang and K. F. Lee, "On Speaker-Independent Speaker-Dependent, and Speaker-Adaptive Speech Recognition," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 2, pp. 877-880, Toronto, Canada, 1991.

[3] J. Godfrey, E. Holliman and J. McDaniel, "Telephone Speech Corpus for Research and Development," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pp. 517-520, San Francisco, California, USA, March 1992.

[4] H. Christensen, "Speaker Adaptation of Hidden Markov Models using Maximum Likelihood Linear Regression," Ph.D. Thesis, Institute of Electronic Systems, Department of Communication Technology, Aalborg University, 1996.

[5] C. J. Leggetter and P. C. Woodland, "Speaker Adaptation Using Linear Regression, Technical Report CUED/F-INFENG/TR.181", University of Cambridge Engineering Department, Cambridge, UK, June 1994.

[6] Y. Grenier, "Speaker Adaptation through Canonical Correlation Analysis," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 3, pp. 888-891, Denver, Colorado, USA, 1980.

[7] Y. Grenier, L. Miclet, J. C. Maurin, and H. Michel, "Speaker Adaptation for Phoneme Recognition, *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 3, pp. 1273-1275, Atlanta, Georgia, USA, 1981.

[8] F. S. Gurgen and H. C. Choi, "On the Frame-Based and Segment-Based Non-linear Spectral Transformation for Speaker Adaptation," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 2, pp. 534-539, Perth, Australia, December 1994.

[9] K. Choukri, G. Chollet, and Y. Grenier,

"Spectral Transformations through Canonical Correlation Analysis for Speaker Adaptation in ASR," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing,* Vol. 4, pp. 2659-2662, Tokyo, Japan, 1986.

[10] F. Kubala, R. Schwartz, and C. Barry, "Speaker Adaptation from a Speaker Independent Training Corpus," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing,* Vol. 1, pp. 137-140, Albuquerque, NM, USA, 1990.

[11] H. C. Choi and R. W. King, "A Two-Stage Spectral Transformation Approach to Fast Speaker Adaptation," *Proceedings of Speech Science and Technology,* Vol. 2, pp. 540-545, Perth, Australia, December 1994.

[12] C. H. Lee, C. H. Lin, and B. H. Juang , "A Study on Speaker Adaptation of Continuous Density HMM Parameters," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 1, pp. 145-148, Toronto, Canada, 1991.

[13] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society,* 39:1-38 Series B, 1977.

[14] C. J. Leggetter, "Improved Acoustic Modeling for HMMs using Linear Transformations," Ph. D. Thesis, University of Cambridge, Cambridge, UK, 1996.

[15] A. J. Hewett, "Training and Speaker Adaptation in Template-based Speech Recognition," Ph.D. Thesis, University of Cambridge, Cambridge, UK, 1989.

[16] Y. Hao, and D. Fang, "Speech Recognition Using Speaker Adaptation by System Parameter Transformation," *IEEE Transactions on Speech and Audio Processing,* Vol. 2, No. 1, Part 1, pp. 63-68, January 1994.

[17] S. Nakamura and K. Shikano, "Speaker Adaptation Applied to HMM and Neural Networks," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 1, pp. 89-92, Glasgow, Scotland, 1989.

[18] P. Zhan, M. Westphal, M. Finke and Alex Waibel, "Speaker Normalization and Speaker Adaptation - A Combination for Conversational Speech Recognition," *Proceedings of Eurospeech* 1997, Rhodes, Greece, September 1997.

[19] M. Gales and P.C. Woodland, "Variance Compensation Within the MLLR Framework," *Technical Report CUED/F-INFENT/TR242,* University of Cambridge Engineering Department, Cambridge, UK, February 1996.

[20] F. Jelinek, *Statistical Methods for Speech Recognition*, The Massachusetts Institute of Technology Press, Cambridge Massachusetts, 1997, pp. 147-163.