Rheinisch Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik VI
Prof. Dr.-Ing. Hermann Ney

Seminar Speech Recognition and Language Processing SS 2003

Speaker Adaptation using Eigenvoices

Daniel Schneider

Matriculation Number 219 860

Date of Presentation: 30.07.2003

Tutor: Michael Pitz

Contents

List of Tables

List of Figures

1 Introduction

In most cases, humans recognize immediately and completely what has been said by their dialog partners. They are even able to recognize words of speakers they do not know at all with almost the same recognition rate.

Human-machine communication differs significantly from that between humans: the performance of automatic speech recognition systems highly depends on the amount of acoustic information which is available from the speaker for training the recognizer. Systems which can be trained by the user before the actual recognition starts generally perform much better than systems that can't obtain this speaker-specific information in a dedicated training phase.

It depends on the application how much - if any - information is available: the user of a dictation system would probably accept a longer training session as the application might be used over a long period of time. The information is gathered a priori, i.e. the speaker trains the system before actually using it for recognition. This is an example of *speaker-dependent* (SD) speech recognition.

But there are applications of speech recognition where information about the speaker is sparse or not available at all. Users of information systems controlled by speech input would not tolerate a long lasting training. A customer who needs information about train timetables and communicates with a timetable information system over a telephone line wants the system to react immediately and reliably to his requests. Another example is the transcription of speech in video and audio media like broadcast news, where the original speaker is not available for training. A general approach to this problem is the use of speaker-independent (SI) systems. SI-recognizers are trained with a large population of speakers by combining their acoustic properties. The system is trained on a broad basis, integrating inter-speaker variabilities. The acoustic input from a speaker during recognition is not used for further improving the system.

Compared to the SI approach, speaker-dependent recognition always performs better as it is trained on a specific user. A possible way to make up this disadvantage is to apply speaker adaptation techniques. Speaker adaptation is the process of modifying an available recognition system and adapting it to a new speaker in order to improve its recognition performance [Giuliani & DeMori 97]. This seminar paper deals with the Eigenvoice approach as a certain method for speaker adaptation.

After introducing the basic concepts and the main categories of speaker adaptation, a general description of the model-based approach - to which the Eigenvoice technique belongs - presents the necessary theoretical basis. An overview on the different types of model-based techniques completes the introductory section. Section 3 is devoted to a detailed description of the Eigenvoice approach. After introducing the technique's goals and its basic mathematical ideas, the paper presents the details of the Eigenvoice adaptation process and the required methods. The presentation of available experimental results in section 3.4 is followed by a short summary and a final conclusion.

2 Approaches to Speaker Adaptation

2.1 Overview

The need for speaker adaptation is closely related to the already mentioned tradeoff between the performance of SD systems and the flexibility of SI approaches.

SD systems create and maintain a specific model for each speaker who uses the recognizer. Each of the users has to train his own model and thus providing speaker-specific information. In contrast to this, SI-systems are usually trained with data from a large set of speakers. The integration of the variability between the training speakers results in a model for a ”mean” speaker which is then used for recognition. Acoustic data from new and unknown speakers during recognition - the *adaptation data* - is not used to improve this model. At this point, speaker adaptation techniques can be applied to combine the training information with the incoming adaptation data from the unknown speaker.

There are several possibilities to subdivide speaker adaptation techniques. A major difference is the way in which the approaches handle the adaptation data from the unknown speaker [Giuliani & DeMori 97].

Supervised or unsupervised. If there is a transcription of the adaptation data - i.e. the actual phrases used for adaptation are known to the system - the adaption is *supervised*. Without this transcription, the adaptation is *unsupervised*. In the latter case, the adaptation data itself has to be recognized before performing the actual adaptation. The recognition of the adaptation data could be carried out with an additional SI-model, for example.

Static or incremental. In *static* speaker adaptation, the adaptation is performed in a dedicated adaptation phase before starting the actual recognition. An example is *two-pass* adaptation, where the data from an unknown speaker is used for adaptation in the first pass and recognized in the second. This approach could be used for recognition of broadcast news, for example. In contrast to this, the dynamic *incremental* speaker adaptation is performed after each new utterance from the unknown speaker, thus improving the recognition performance as more adaptation data becomes available.

Moreover, speaker adaptation techniques can be divided into two main categories: feature vector transformation and model-based adaptation.

The first step in both training and recognition is to convert the continuous acoustic input signal from a speaker into discrete feature vectors as described in [Ney 03, p. 34-45]. Feature vector transformation or normalization tries to estimate a transformation from the feature space of the new speaker into the feature space of a training speaker in order to reduce the acoustic differences like different sex or age. The transformation can be calculated with a small amount of adaptation data. The feature vectors obtained in the actual recognition are then transformed into the feature space of the training speaker. Thus a well-trained model can be applied.

In contrast to the adaptation in the feature space, it is possible to perform the adaptation in the model parameter space. To characterize a training speaker, recognition systems estimate an acoustic model from the feature vectors of the training data (see section 2.2). This model contains parameters that represent the acoustic properties of the speaker, which may vary according to his sex, age, or origin, for example. These parameters can be adapted by speaker adaptation techniques to customize the model for an unknown speaker. The model-based approach uses a small amount of adaptation data from the new speaker to adapt the parameters of the trained model. This type of speaker adaptation will be described in more detail in section 2.2 as the Eigenvoice approach is based on model adaptation.

2.2 Model-Based Approaches

A common approach for modelling acoustic structures like words or phonemes is the use of Hidden Markov Models (HMMs). This section introduces this modelling technique and derives a set of parameters to characterize an HMM. These parameters - well-trained by training speakers before the actual recognition - will be adapted by model-based speaker adaptation to an unknown user.

A HMM is used to model a specific acoustic phoneme. A whole word is then modelled by concatenating the HMMs of the corresponding phonemes [Ney 03, pp. 183]. In the following, isolated word recognition with word-level HMMs will be considered, i.e. the input speech corresponds to one word which is modelled by one HMM.

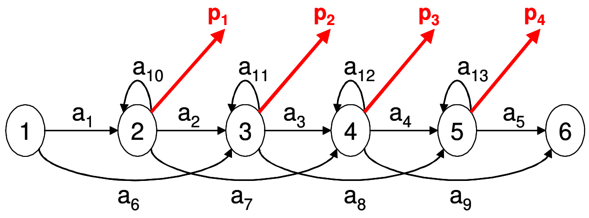

Figure 1: Hidden Markov Model

An HMM for a certain word w is defined as a graph which consists of several nodes (*states*) and arcs (*transitions*) as illustrated in figure 1, modelling the structure of the word. The graph has one *initial* and one *final* state (states *1* and *6* in figure 1), the remaining states are called *internal*. A *transition probability* is associated to every transition of the graph (a_i in figure 1). Moreover, an *emission probability* p is associated to every internal state which is then called *emitting state* [Westwood 99]. While the transition probability $a = p(s,s')$ indicates how probable the transition between the two connected states s and s' is, the emission probability $p = p(x \mid s,w)$ shows how probable state s of the modelled word w produces a certain acoustic output x.

During recognition, the automatic speech recognition (ASR) system receives acoustic input from the speaker and converts the continuous signal into T feature vectors x_t, each of dimension D. For each time frame t in the sequence $1,\dots,T$, the system calculates this high dimensional vector x_t that characterizes the corresponding time frame in the continuous signal. The ASR system compares the sequence of feature vectors $x_1^T = x_1 \cdots x_T$ to all available HMMs and tries to find the HMM with the smallest distance to the speech input.

This distance can be modelled with $p(x_1^T \mid w)$, the probability that an input sequence x_1^T corresponds to a certain HMM modelling the word w. The higher this probability, the better the match between HMM and spoken word. The word that is modelled by the HMM with the highest probability is finally recognized. To calculate this probability, the systems sums up the probabilities of all paths $s_1^T = s_1 \cdots s_T$ with length T through the current HMM that start at the initial state and end at the final state:

$$p(x_1^T \mid w) = \sum_{[s_1^T]} p(x_1^T, s_1^T \mid w) \tag{1}$$

The probability of a single path $p(x_1^T, s_1^T \mid w)$ consists of the emission and transition probabilities illustrated in figure 1. As the emission probability p models the acoustic differences between speakers, it highly depends on the speaker. Hence the parameters of the underlying distribution can be adapted by speaker adaptation to improve the recognition performance. The emission probability is usually modelled with a gaussian distribution:

$$p(x \mid s,w) \quad = \quad \frac{1}{\sqrt{(2\pi)^D \|\Sigma_{sw}\|}} \exp^{-\frac{1}{2}(x-\mu_{sw})^T \Sigma_{sw} (x-\mu_{sw})} \tag{2}$$

where μ_{sw} is the mean and Σ_{sw} the covariance matrix of the gaussian distribution corresponding to state s of word w and D is the dimension of the feature vector x. With this assumption, it is sufficient to store the parameters of the gaussian distribution for every state s of every word w: the means μ_{sw} and the covariance matrices Σ_{sw} make up a highly speaker-dependent part of the model for a specific speaker.

Using multimodal gaussian distributions, a set of mixtures is assigned to every state. Assuming that the states of all HMMs are ordered and labeled with $s = 1, ..., S$ and that the gaussian mixtures for each state s are labeled with $l = 1, ..., L_s$, equation 2 becomes

$$p(x \mid s) = \sum_{l=1}^{L_s} p(l \mid s) p(x \mid l, s) \quad (3)$$

$$p(x \mid l, s) = \frac{1}{\sqrt{(2\pi)^D \|\Sigma_l^s\|}} \exp^{-\frac{1}{2}(x-\mu_l^s)^T \Sigma_l^s (x-\mu_l^s)} \quad (4)$$

The model parameter set is extended with the corresponding *mixture weights* $p(l \mid s)$ as the probability for component l in state s. The covariance matrices Σ_l^s and the means μ_l^s have to be stored for every component l of every state s.

Thus a parameter set $\lambda = \left\{\mu_l^s, \Sigma_l^s, p(l \mid s)\right\}$ for all l and s can be used to characterize a speaker for speaker adaptation purposes. In principle, the ASR system is usually trained with a maximum likelihood (ML) estimation where the likelihood of the training data $p(x \mid \lambda)$ is maximized over a parameter set λ. The parameter set with the highest likelihood is then chosen for the model, see [Ney 03], [Giuliani & DeMori 97] or [Faltlhauser 02] for detailed descriptions. In speaker-dependent systems, it is the speaker that uses the system for recognition who trains the system, resulting in a model with a speaker-specific parameter set λ. In SI-recognizers, a population of several training speakers performs the training and a mean of the resulting models is then used for the recognition process, covering the inter-speaker variability of the training population.

The general idea of model-based speaker adaptation is to adapt the parameters λ of a well-trained model to a new speaker during the adaptation process. Traditionally, adaptation is applied to a SI-model, but some approaches - like Eigenvoices - take SD-models as a basis, see section 3.

The different model-based approaches can be subdivided into three families which are informally described below.

2.2.1 Maximum a posteriori

The maximum a posteriori parameter estimation (MAP) re-estimates the parameter set λ of a distribution, i.e. the distribution corresponding to a trained SI model, by using the parameter's prior distribution $p(\lambda)$. This prior distribution can be estimated during training. With available adaptation data x_1^T from an unknown speaker, the parameter set λ is adapted, resulting in a new parameter set $\tilde{\lambda}$:

$$\tilde{\lambda} = \underset{\lambda}{\operatorname{argmax}}\, p(\lambda \mid x_1^T) = \underset{\lambda}{\operatorname{argmax}}\, p(\lambda) p(x_1^T \mid \lambda) \quad (5)$$

$p(x_1^T \mid \lambda)$ is the prior distribution of the adaptation data [Faltlhauser 02, page 20]. There are several possibilities for estimating the new parameters, [Woodland 01] provides a short introduction, re-estimation formulae for λ can be found in [Faltlhauser 02].

MAP is a local approach as it only adapts parameters of HMMs seen in the adaptation data. With little data, only small changes are applied to the model as only a few HMM parameter sets are updated. If more data becomes available, the adapted model converges to the ML estimate mentioned in section 2.2. With sparse adaptation data, MAP leaves a major part of the parameters unaltered. This is problematic for applications where rapid speaker adaptation is required and only little adaptation data is available. Moreover, MAP converges rather slow to the ML estimate.

There are several extensions to the MAP approach that try to solve these problems [Woodland 01, page 12].

2.2.2 Linear Transformation

A second approach is to estimate a linear transformation for the parameter set of the trained model with information from adaptation data and thus adapting the model to an unknown speaker. The goal is to find a transformation that can be applied to a major part of the parameters without having observed all corresponding words in adaptation data. Parameters of models not seen in adaptation data are modified along with similar parameters which have been observed [Faltlhauser 02, page 21]. Maximum likelihood linear regression (MLLR) is a commonly used representative of this family and is therefore described below.

MLLR uses an affine transformation to update the gaussian mean parameters. The variances and mixture weights can be obtained from an SI-model, for example. A transformation matrix A and a translation b are estimated from adaptation data. A gaussian mean is adapted by

$$\tilde{\mu} = A\mu + b = W\mu_{ext} \tag{6}$$

where $W = [bA]$ is a matrix for the affine transformation and $\mu_{ext}^T = \left[1\mu^T\right]$ is the extended mean vector. W can be estimated using Expectation-Maximization (EM) [Woodland 01], see section 3.3.2 for a general introduction to this algorithm. As a transformation matrix can be tied over several gaussian parameters, the idea of adapting parameters of unseen HMMs along with those of seen models can be applied in MLLR: a matrix calculated with the information of very little adaptation data can update a large part of the model parameters. If more adaptation data becomes available, more transformation matrices can be estimated, each updating a certain part of the whole model.

MLLR updates all - or at least the majority - of the model parameters even if only little adaptation data is available. But for very sparse data, i.e. a few seconds or a single word, the applied global transform is rather imprecise [Woodland 01].

2.2.3 Speaker Clustering

The third family of model-based speaker adaptation is speaker clustering. Traditionally, this approach clusters the speakers of the training population and estimates HMMs representing the clusters. One of the cluster models is selected for recognition with the information from adaptation data (usually this decision is not made until the end of the recognition process, where the cluster model with the best performance is chosen). The most simple approach is to obtain the clusters with an a priori decision, for example by distinguishing between male and female speakers and building a model for each gender. This approach can be improved by increasing the number of clusters and combining certain other types of speakers (according to their age, origin, etc...). But the a priori decision about the types of clusters that should be constructed can lead to fragmented training data: if the choice does not match the structure of the reference population, the training data is fragmented, i.e. some clusters receive loads of data, others only little. The process of selecting a cluster structure can be automated to match the structure of the training population. But in both cases, the unknown speaker is assigned to only one cluster. This is problematic as the mentioned training data problem does not allow to build a lot of clusters for a high number of different speaker types.

To overcome this, the basic clustering principle can be improved by representing the unknown speaker as a weighted combination of all available cluster models. If only the gaussian mean should be adapted, the idea of combining the clusters can be formalized with

$$\tilde{\mu} = \sum_{c=1}^{C} w_c \mu_c \tag{7}$$

where μ_c is the gaussian mean of cluster c, w_c is the weight for that cluster and $\tilde{\mu}$ is the adapted mean for the new speaker. The C clusters form a space that contains all possible models, each represented by a different weighted combination of the clusters. The more a speaker belongs to a certain cluster, the higher is the effect of that cluster model on the adapted model. Thus a model for the new speaker is built by finding the weights for the cluster combination with the information from adaptation data.

Cluster Adaptive Training (CAT) as published by Gales in [Gales 00] is based on this idea. The Eigenvoice technique is quite similar and uses comparable methods for the adaptation itself once the initial cluster space is estimated (section 3.5 discusses the difference between CAT and the Eigenvoices). But in the Eigenvoice approach, a speaker is rather represented by a weighted sum of *directions of variation* than by a sum of *models*. The idea will be presented in more detail in the following sections.

3 Eigenvoices

3.1 Faces and Voices

The Eigenvoice technique was first published by Kuhn et al. in [Kuhn & Nguyen$^+$ 98]. Looking for a rapid adaptation technique to handle the sparse data problem, Kuhn took up ideas already used in human face recognition. There is a certain similarity between the automatic recognition of speech and the recognition of human faces: although the parameter space that makes up the characterization of speech is as high dimensional as that of images, one might think intuitively of only a few parameters that are sufficient to represent a speaker or a human face [Kirby & Sirovich 90].

The term *Eigenvoices* is derived from a technique used in face recognition called *Eigenfaces*. Generally speaking, face recognition is the task of finding the image of a human face in a database of 2-D human face images. The idea of the Eigenface approach is that the differences between the 2-D images in the database and their common properties can be described using very few parameters, i.e. the dimensionality of the *face space* is rather low. A technique for estimating a certain mathematical representation of these variations is introduced in [Kirby & Sirovich 90]: Eigenfaces. Each Eigenface calculated from of a population of face images represents a certain property of these faces: the first Eigenface covers the variation of the hairstyle, the second represents the size of the face and so on [Oliva 03]. The face recognition system tries to find a face in the database by representing the new face as a linear combination of the Eigenfaces: for example, the first Eigenface is weighted depending of the unknown face's hairstyle. Then the distance between this representation of the new face and the low dimensional representation of every face in the database is calculated. The image with the smallest distance is finally recognized. The number of parameters that have to be compared is greatly reduced: while traditional approaches compare whole images, i.e. their high dimensional representation, the Eigenface technique only has to compare the linear combination of the new face to the low dimensional representations of the faces in the database. Figure 2 shows an attempt to visualize the first few Eigenfaces derived from a database of about 500 face images [MIT 03].

But how does this approach correspond to speaker adaptation? At first glance, it is rather suitable for speaker identification than for speaker adaptation.[1] As mentioned in section 2.1, tradi-

[1]Indeed there is ongoing research on this topic using Eigenvoices, see [Thyes & Kuhn$^+$ 00]

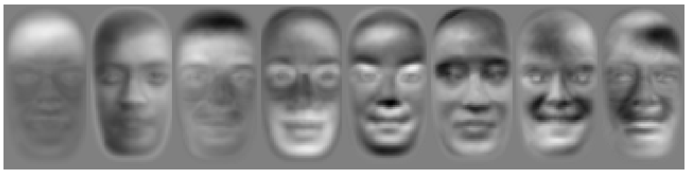

Figure 2: Visualization of the first few Eigenfaces

tional speaker adaptation is the modification of a trained SI model to improve the recognition performance with an unknown speaker. The Eigenvoice approach goes a step further: every speaker can be represented by a weighted sum of only a few *Eigenvoices* that correspond to the Eigenfaces mentioned above. Thus the Eigenvoices span a low-dimensional space that contains virtually all possible SD models, the *speaker space*. An appropriate model for the new speaker is located somewhere in this speaker space. Hence the goal of the Eigenvoice technique is to construct the speaker space from the training speakers and to find the location of the model representing the unknown speaker. Both parts of the approach will be described in detail in section 3.2 and 3.3, respectively.

Although the characteristics of human speech are not as obvious as those of human faces, some properties modelled by the Eigenvoice vectors are quite illustrating for understanding the approach: the first Eigenvoice corresponds to the gender of the speaker while the second contains information about the amplitude of the speech.

The Eigenvoice technique consists of two parts: the estimation of the low-dimensional speaker space and the estimation of the new speaker's model. The construction of the speaker space by estimating the Eigenvoices is carried out offline and only once, i.e. before any speaker can actually use the system for recognition. Hence the computational load of the methods described in section 3.2 does not affect the recognition process itself. Estimating the new speaker's model is an online step carried out with every unknown speaker. Figure 3 gives a first illustration of the whole process that is described in detail in the following subsections.

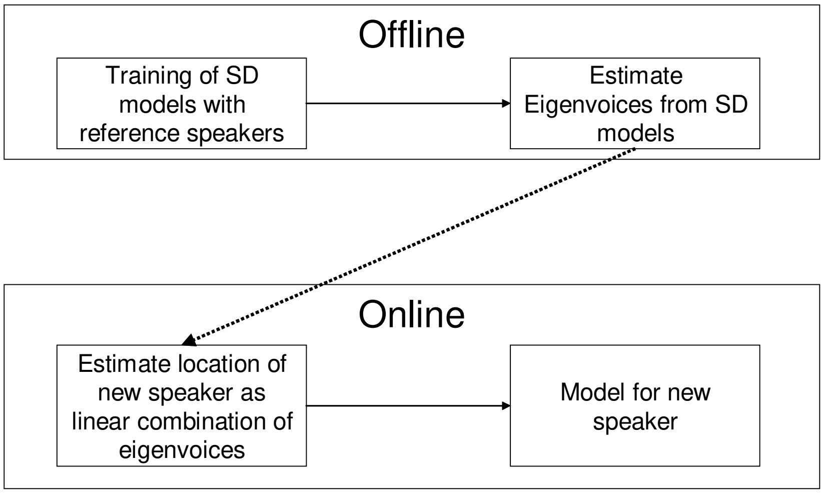

Figure 3: Illustration of the Eigenvoice technique

3.2 Offline: Estimating the Speaker Space

Before actually starting with the recognition part, the speaker space is estimated. As already mentioned, the idea is to estimate a space that contains virtually all possible SD models.

A set of well trained SD models is the starting point for the estimation of the space. The parameters of these models build the *parameter space*. If a model consists of D one-dimensional gaussian means, then the parameter space is of dimension D, too. Usually, the means are high dimensional as well, as they have the same dimension as the feature vectors derived from the acoustic signal which typically cover 33 features. This results in a parameter space of even higher dimension.

As models usually contain numerous HMMs, and each HMM is characterized by several gaussian means, it is clear that the parameter space is rather high dimensional. The Eigenvoice technique reduces the dimension of the problem by estimating a subspace of the parameter space which is of lower dimension but still covers the variation between the trained models: the already mentioned speaker space. This space is spanned by the Eigenvoice vectors. Each Eigenvoice vector covers a certain direction of variation between the models of the training speakers. Figure 4 illustrates the necessary steps that are described below.

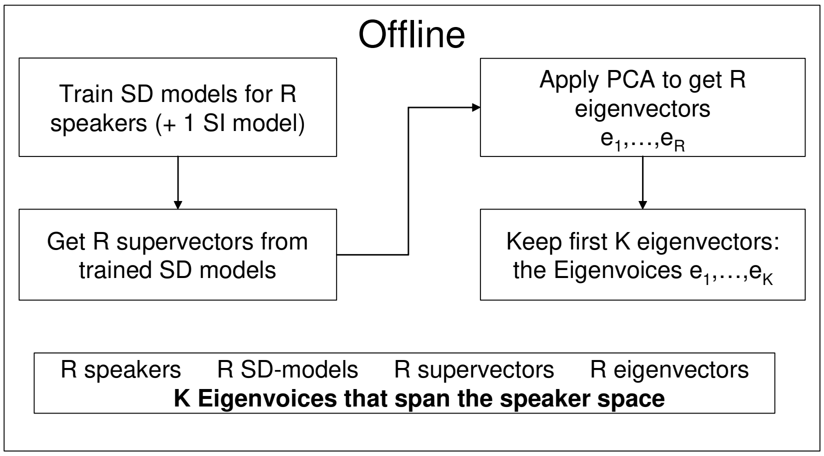

Figure 4: Required steps for estimating the Eigenvoices, parts from [Kuhn & Nguyen$^+$ 00, p. 696, Fig.1]

Suppose there a R training speakers to build the initial models. The first step is to train their SD models, for example with the maximum likelihood estimation mentioned in section 2.2. Moreover, it is necessary to train an additional SI model that is required for additional HMM parameters in a later step: only the gaussian mean is adapted to the new speaker, the variance and the gaussian mixture weights have to be obtained from the SI model [Woodland 01].

For the mathematical requirements of the following steps, the models' gaussian mean values have to be arranged in the form of a vector, the so-called *supervector*. A supervector for a certain training speaker contains all gaussian means representing all HMMs trained by that particular speaker. Thus the vector is of the same dimension as the parameter space, D. For every speaker $r \in \{1,...,R\}$ the supervector $\hat{\mu}_r$ is built in the following way [Kuhn & Nguyen$^+$ 98]:

1. Collect all gaussian means from all components of all states trained by speaker r: $\{\mu_l^s(r) \mid l \in \{1,...,L_s\}, s \in \{1,...,S\}\}$

2. Put the means into a vector, the supervector $\hat{\mu}_r$

3. The order in which the means are put into the vector is arbitrary but has to be the same for all R supervectors.

$$\tilde{\mu}_r = \begin{bmatrix} \tilde{\mu}_1^1(r) \\ \vdots \\ \tilde{\mu}_l^s(r) \\ \vdots \\ \tilde{\mu}_{L_S}^S(r) \end{bmatrix} \tag{8}$$

This results in a set of R supervectors $\{\hat{\mu}_1, ...\hat{\mu}_R\}$, each of high dimension D: each supervector contains the parameters of all HMMs trained by a speaker, each HMM contains several states with several mixtures, each with a certain gaussian mean. Finally, each of these means is high dimensional as well. Equation 8 shows the structure of a supervector for a particular training speaker r.

After the construction of the R supervectors, an orthogonal basis of the speaker space is estimated. This is achieved by rotating the axes of the parameter space along the main directions of variability. Figure 5 illustrates the process applied to a two-dimensional parameter space: The original space contains the parameters μ_1 and μ_2 derived from a two-state HMM without using mixture densities. Each point corresponds to a pair $(\mu_1(r), \mu_2(r))$ representing the trained model of a speaker r. The figure visualizes the trained models of 30 different speakers. The vectors e_1 and e_2 form a orthogonal basis of the rotated space, and each pair (μ_1, μ_2) can be represented as a linear combination of e_1 and e_2:

$$\hat{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} = w_1 e_1 + w_2 e_2 \tag{9}$$

w_1 and w_2 are weights that depend on the speaker that is represented by the pair of gaussian means. With this formalism, it is possible to describe *all* possible models as a linear combination of the basis vectors, i.e by the two corresponding weights w_1 and w_2.

The dimensionality reduction is achieved by applying principal component analysis (PCA): it performs the described rotation while ordering the new basis vectors along their contribution to the variation between the parameters [Jolliffe 86]. PCA is used for reducing the dimensionality by Kuhn et. al, but they remark that other techniques like linear discriminant analysis or independent component analysis might be applied as well [Kuhn & Nguyen$^+$ 98]. In figure 5, the main variation direction is covered by e_1, the orthogonal e_2 only has a minor influence on the overall variability. A key idea of the Eigenvoice technique is that a model can be adequately represented by a combination of only a few of the above mentioned basis vectors, as long as they cover most of the variation between the training speakers. In the case of the two-dimensional example, it could be sufficient to restrict the pair of gaussian means to lie on the line spanned by e_1.

PCA calculates the eigenvectors from the covariance matrix of the supervectors to estimate a basis of the rotated space. For R supervectors, the covariance matrix is defined as

$$\Sigma_{sup} = \frac{1}{R} \sum_{r=1}^{R} (\hat{\mu}_r - \hat{\mu})(\hat{\mu}_r - \hat{\mu})^T \tag{10}$$

Σ_{sup} is a $D \times D$-dimensional symmetrical matrix with $rank(\Sigma_{sup}) \leq R$ as there are only $R \cdot D$ different observations to fill the $D \cdot D$ entries of Σ_{sup}, resulting in at least $(D - R)$ linear dependent rows. Thus applying PCA to Σ_{sup} yields not more than R different eigenvalues, sorted in descending order. In the following, it is assumed that PCA produces R eigenvalues. The eigenvectors

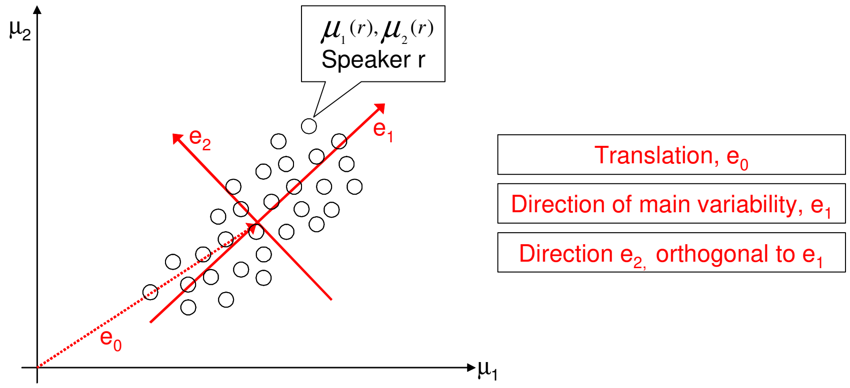

Figure 5: Main axes of two-dimensional distribution

$e_1, \ldots, e_R$ corresponding to the sorted eigenvalues span the rotated space, where e_1 belongs to the largest and e_R to the smallest eigenvalue. Efficient algorithms to find the eigenvalues as well as implementation issues are covered in [Jolliffe 86]. An alternative is to use the correlation matrix of the supervectors instead of the covariance matrix. The rotation along the directions of variation is more exact because the correlation matrix does not allow large absolute values among the components of the supervector to dominate the PCA [Kuhn & Nguyen$^+$ 98].

The eigenvectors are ordered along the size of the corresponding eigenvalues, and hence ordered along their contribution to the variation. Kuhn et. al found that only a few of the eigenvectors are sufficient to cover a major part of the overall variation between the supervectors. Keeping only the first K eigenvectors still covers most of the variation between the supervectors and results in an immense reduction of the dimensionality as the sum of parameters decreases with $K < R \ll D$:

$$K \cdot D < R \cdot D \tag{11}$$

These first K eigenvectors are called *Eigenvoices*. Each Eigenvoice e_j for $j = 1, \ldots, K$ illustrated by figure 12 has the same dimension and structure as the supervectors described above. $e_l^s(j)$ is the subvector of the jth Eigenvoice e_j corresponding to component l of state s.

$$e_j = \begin{bmatrix} e_1^1(j) \\ \vdots \\ e_l^s \\ \vdots \\ e_{L_S}^S(j) \end{bmatrix} \tag{12}$$

Kuhn et al. experimented with different values for K, their results are shown in section 3.4.1. Typical values for K are rather small ($K \leq 10$) compared to $R = 120$ trained SD models used in the experiment. D, the dimension of the supervector, varies depending on the type of the recognition task. If a small vocabulary is used, then D is rather small as well: in the Isolet task described in section 3.4.1, there are 26 HMMs with 6 states each, resulting in 156 gaussian means of dimension 18 per training speaker. Thus the supervector is of dimension $D = 156 \cdot 18 = 2808$. In tasks with a large vocabulary, the dimension of the supervector can reach $D = 10^6$ [Botterweck 00].

The high dimension can result in both storage and numerical problems when performing principal component analysis: in Botterweck's experiments, the covariance matrix that has to be diagonalized for finding the eigenvalues is of dimension $1.000.000 \times 1.000.000$. Moreover, the training data available from training speakers of large vocabulary corpora is usually not sufficient to train a SD-model for each of the speakers. Botterweck solves this by training an SI-model with the data from all training speakers. The SD-models of the speakers required for PCA are obtained by adapting the SI-model to every speaker using MAP or MLLR, yielding different models for all different training speakers. The time complexity of diagonalizing the matrix does not affect the recognition itself as the eigenvectors have to be found before the actual recognition and only once (Botterweck reports 1,5 hours on a 500MHz Alpha CPU in [Botterweck 00]). Botterwecks experimental results are presented in section 3.4.2.

3.3 Online: Locating the Unknown Speaker

3.3.1 Overview

This and the following section describe the second part of the Eigenvoice approach: the location of the unknown speaker within the speaker space.

If an unknown speaker U would perform an intense training, the resulting SD-model - or in terms of Eigenvoices: his supervector $\hat{\mu}_U$ - would be an element of the rotated D-dimensional parameter space. It could be represented by a linear combination of all R eigenvectors derived from the covariance matrix of the R supervectors estimated from the training speakers. With the assumption that the first K eigenvectors cover most of the variation between the speakers in the training population (and thus virtually covering the variation between *all* speakers if the population is well chosen), a sufficient model $\tilde{\mu}_U$ for U can be considered to lie in the K-dimensional speaker space. As the Eigenvoices build an orthogonal basis of the speaker space, the adapted model $\tilde{\mu}_U$ is represented by a linear combination of the Eigenvoices:

$$\tilde{\mu}_U = \sum_{j=1}^{K} w_j \cdot e_j \tag{13}$$

With this assumption, it is sufficient to estimate the weights w_j for adapting a model to the unknown speaker. The coefficients of the Eigenvoices are estimated with the information available from adaptation data $x_1^T = x_1, ..., x_T$.

Before estimating the weights as described below the observations x_1^T are *normalized* by subtracting the "mean" of the trained SD-models e_0 from each observation $x_t, t = 1, .., T$. This reflects the translation of the original parameter space by e_0, see figure 5.

Kuhn et al. have developed and applied two methods for estimating the Eigenvoice coefficients: Projection and maximum likelihood eigen-decomposition (MLED). Projection is the original method used for the corresponding Eigenface task [Nguyen 98, page 31]. A SD-model is trained with the sparse adaptation data from the unknown speaker U. A supervector $\hat{\mu}_U$ is built from the model in the same way as described in section 3.2. This vector is then projected from the original D-dimensional parameter space into the K-dimensional speaker space, yielding the K weights $w_1, ..., w_K$ and thus the adapted model $\tilde{\mu}_U$ for the unknown speaker as defined by equation 13. The speaker space is a subspace of the parameter space and its orthogonal basis is defined by the Eigenvoices $e_1, ..., e_K$, so the projection can be performed using standard methods from linear algebra [Höllig & App 03].

As the technique relies on a SD-model trained by the unknown speaker, it cannot be used for fast speaker adaptation where only a small amount of adaptation data is available to train the initial model: a major part of the HMMs are not seen in the short adaptation phase and can therefore not be estimated. The projection method has been extended by Westwood using a transformation that weights the parameters of the SD model before projecting: parameters of HMMs seen in adaptation data receive a higher weight, unseen parameters have a smaller influence. Projection is no longer used by Kuhn et al. [Kuhn & Nguyen+ 00, p. 697], they concentrate on MLED which is described below.

3.3.2 Maximum Likelihood Eigen-Decomposition

MLED, introduced in [Nguyen 98], iteratively estimates a set of weights $\tilde{w}_1, ..., \tilde{w}_K$ that build an adapted supervector $\tilde{\mu}_U$ for the unknown speaker U. Thus the supervector for the speaker is restricted to lie in the speaker space spanned by the Eigenvoices $e_1, ...e_K$. The goal is to maximize the likelihood $\mathcal{L}(\lambda \mid x_1^T)$ of a parameter set $\lambda = \{w_1, ..., w_K\}$ regarding the adaptation data x_1^T.

$$\mathcal{L}(\lambda \mid x_1^T) = p(x_1^T \mid \lambda) = \prod_{t=1}^{T} p(x_t \mid \lambda) \tag{14}$$

Usually, the log-likelihood is used instead:

$$log\mathcal{L}(\lambda \mid x_1^T) = logp(x_1^T \mid \lambda) = \sum_{t=1}^{T} logp(x_t \mid \lambda) \tag{15}$$

The log-likelihood is maximized over λ, resulting in a parameter set $\tilde{\lambda} = \{\tilde{w}_1, ..., \tilde{w}_K\}$:

$$\tilde{\lambda} = \underset{\lambda}{\operatorname{argmax}} \sum_{t=1}^{T} logp(x_t \mid \lambda) \tag{16}$$

The supervector $\tilde{\mu}_U = \sum_{j=1}^{K} \tilde{w}_j \cdot e_j$ build with the thus obtained $\tilde{\lambda}$ contains adapted gaussian means for all states of all HMMs trained by the training population. The means are used for recognition in conjunction with the covariance matrices and mixture weights obtained from the SI model already trained in the offline step. Figure 6 illustrates the adaptation process.

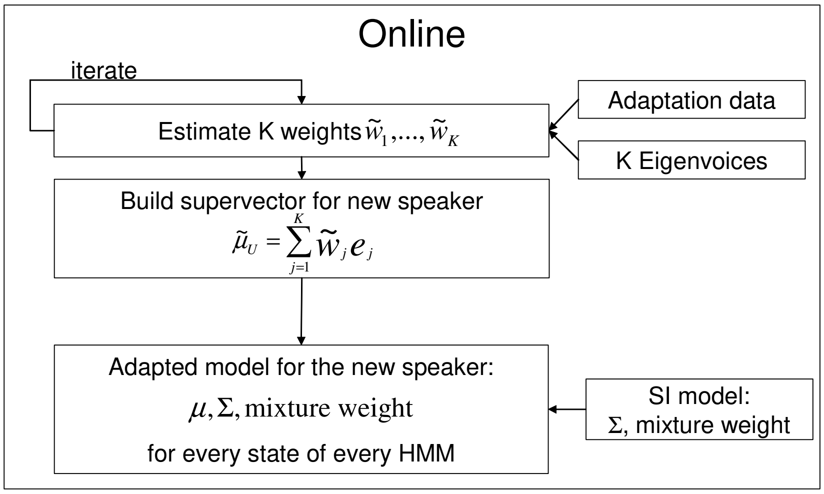

Figure 6: Required steps for MLED, parts from [Kuhn & Nguyen+ 00, p. 696, Fig.1]

The Problem. The probability that an observation x_t of the adaptation data x_1^T belongs to a gaussian mixture component l of a state s is unknown. Without this *occupation probability* $\gamma_l^s(t)$, the adaptation data is *incomplete* and the gaussian means for a certain component of a certain state cannot be estimated directly.

The Expectation-Maximization solution. The iterative estimation of the weights is based on the Expectation-Maximization (EM) algorithm. The missing information can be modelled with a discrete random variable, the *hidden variable* y. All possible values for y are taken into account for every single observation x_t:

$$p(x_t \mid \lambda) = \sum_y p(x_t, y \mid \lambda) = \sum_y p(y \mid \lambda) \cdot p(x_t \mid y, \lambda) \tag{17}$$

With this assumption, the log-likelihood is optimized over λ by

$$\begin{aligned} \tilde{\lambda} &= \underset{\lambda}{\operatorname{argmax}} \left\{ log \mathcal{L}(\lambda \mid x_1^T) \right\} && (18) \\ &= \underset{\lambda}{\operatorname{argmax}} \sum_{t=1}^{T} log p(x_t \mid \lambda) && (19) \\ &= \underset{\lambda}{\operatorname{argmax}} \sum_{t=1}^{T} log \left(\sum_y p(x_t, y \mid \lambda) \right) && (20) \end{aligned}$$

The distribution of y is unknown but required for maximizing the likelihood. The EM algorithm solves this by first *expecting* y from an initial parameter set λ and then *maximizing* over a current set $\tilde{\lambda}$ with the expected values for y, yielding an updated parameter set $\tilde{\lambda}$. These two steps are iterated with $\lambda = \tilde{\lambda}$ as the initial parameter set until a convergence criterion is satisfied.

An auxiliary function $Q(\lambda, \tilde{\lambda})$ is used for the two steps of each iteration, where λ is the parameter set estimated in the last iteration and $\tilde{\lambda}$ the set that is maximized in the current iteration.

$$Q(\lambda, \tilde{\lambda}) = \sum_{t=1}^{T} \sum_y p(y \mid x_t, \lambda) \cdot log p(x_t, y \mid \tilde{\lambda}) \tag{21}$$

The expectation step yields the required values for the hidden variable. Q can be maximized over $\tilde{\lambda}$ and the maximizing argument is used for expectation in the next iteration:

$$\lambda = \underset{\tilde{\lambda}}{\operatorname{argmax}} Q(\lambda, \tilde{\lambda}) \tag{22}$$

It can be proven that the likelihood of the parameter set $\tilde{\lambda}$ increases with every iteration of the EM algorithm and that it converges to a local maximum of the likelihood function [Ney 02].

From EM to MLED. The MLED approach makes use of the iterative optimization provided by the EM algorithm. Its goal is to iteratively estimate the adapted gaussian means for the new speaker U in the form of a D-dimensional supervector $\tilde{\mu}_U$ as a weighted sum of the Eigenvoices $e_1, \ldots, e_K$.

Each Eigenvoice vector e_j, $j = 1, \ldots, K$, that has been estimated in section 3.2 is of the same dimension and structure as the supervectors. Equation 25 illustrates the linear combination, using multimodal gaussian distributions with components $l = 1, \ldots, L_s$ for each state s of an HMM. Equation 25 assumes that the states of all HMMs trained by the training population are ordered and labeled with $s = 1, \ldots, S$.

$$\tilde{\mu}_U = \begin{bmatrix} \tilde{\mu}_1^1(U) \\ \vdots \\ \tilde{\mu}_l^s(U) \\ \vdots \\ \tilde{\mu}_{Ls}^S(U) \end{bmatrix} \tag{23}$$

$$= \sum_{j=1}^{K} \tilde{w}_j \cdot e_j \tag{24}$$

$$= \tilde{w}_1 \cdot \begin{bmatrix} e_1^1(1) \\ \vdots \\ e_l^s(1) \\ \vdots \\ e_{Ls}^S(1) \end{bmatrix} + \cdots + \tilde{w}_j \cdot \begin{bmatrix} e_1^1(j) \\ \vdots \\ e_l^s(j) \\ \vdots \\ e_{Ls}^S(j) \end{bmatrix} + \cdots + \tilde{w}_K \cdot \begin{bmatrix} e_1^1(K) \\ \vdots \\ e_l^s(K) \\ \vdots \\ e_{Ls}^S(K) \end{bmatrix} \tag{25}$$

The subvector $\tilde{\mu}_l^s(U)$ of the adapted supervector as well as $e_l^s(j)$ of the j-th Eigenvoice correspond to the gaussian mean of mixture l in state s. Thus, the adapted gaussian mean $\tilde{\mu}_l^s(U)$ for a certain component l of a state s can be expressed as a linear combination of the corresponding eigenvoice subvectors:

$$\tilde{\mu}_l^s(U) = \sum_{j=1}^{K} \tilde{w}_j \cdot e_l^s(j) \tag{26}$$

The auxiliary function for MLED is given in [Nguyen 98]:

$$Q(\lambda, \tilde{\lambda}) = -\frac{1}{2} p(x_1^T \mid \lambda) \sum_{t=1}^{T} \sum_{s=1}^{S} \sum_{l=1}^{L_s} \left(\gamma_l^s(t) \left[-n \cdot log 2\pi - log|\Sigma_l^s| + (x_t - \tilde{\mu}_l^s)^T \Sigma_l^s (x_t - \tilde{\mu}_l^s) \right] \right) \tag{27}$$

The covariance matrices Σ_l^s are obtained from the additional SI model trained in the offline step, n is the dimension of the feature vectors x_t. The occupation probability $\gamma_l^s(t)$ that indicates how probable an observation x_t from adaptation data is mapped to the mixture component l of state s is calculated in the expectation step with the old parameter set λ from the last iteration. In the first iteration, the initial parameter set λ is obtained from the SI model.

The actual maximization which determines the set of weights that maximize equation 27 is carried out as follows: First, the $\tilde{\mu}_l^s$ are replaced by their respective linear combination according to equation 26. Then, the derivative of $Q(\lambda, \tilde{\lambda})$ with respect to any $\tilde{w}_j$, $j = 1, ..., K$ is set to zero to find a local maximum of Q regarding w_j:

$$Q_j = \frac{\partial Q}{\partial \tilde{w}_j} \tag{28}$$

$$= \sum_{t=1}^{T} \sum_{s=1}^{S} \sum_{l=1}^{L_s} \gamma_l^s(t) \left[(-e_l^s(j))^T \Sigma_l^{s-1} x_t + \sum_{k=1}^{K} \tilde{w}_k \, (e_l^s(k))^T \, \Sigma_l^{s-1} \, (e_l^s(j)) \right] \tag{29}$$

$$\stackrel{!}{=} 0 \tag{30}$$

Each of the resulting K equations Q_j has K unknown variables $w_1, ..., w_K$ which can be found by solving the equation system with K equations and K unknown variables.

In every step of MLED, a new model with adapted gaussian means and occupation probabilities is obtained by estimating the K weights. The iteration goes on until the weights converge. In the experiments presented in section 3.4.1, only two iterations of MLED were required.

3.4 Experimental Results

3.4.1 Small Vocabulary, Isolated Words

This section will present the performance of the Eigenvoice approach applied to small vocabulary speech recognition of isolated words. The experiments underlying this section have been carried out by Kuhn and Nguyen, published in [Kuhn & Nguyen$^+$ 00], [Kuhn & Nguyen$^+$ 98] and [Nguyen 98].

Regarding suitable applications for rapid speaker adaptation like telephone dialogues, Kuhn et al. chose a recognition task that is closely related to the requirements of these systems: the recognition of isolated letters of the alphabet. For example, this could be useful for a a system that can be controlled over a telephone line, performing actions according to the letter spoken by the user. The Isolet database [CSLU 03] has been used for both training and testing.

Isolet contains 5 sets of 30 speakers, each of the speakers pronounces all letters of the alphabet twice. The letters are spoken isolated. The data has been split into five evaluation sets: each evaluation set contained all five Isolet sets, but each contained a different Isolet set marked as test data while the remaining four sets were used for training. The experiments have been carried out on all evaluation sets. The results presented below are the average of the results from the evaluation sets.

For the offline training, i.e. the estimation of the speaker space, 120 SD models were constructed with ML estimation using the four sets marked for training. Each of the models contained 26 HMMs (one per letter), each of the HMMs had six states, resulting in 156 states per model. For each state, the 18-dimensional output gaussian has to be stored. Thus the dimension of the $R = 120$ constructed supervectors was $S = 156 * 18 = 2808$, representing the SD models for the training speakers and containing all required parameters. Then, PCA was applied on the correlation matrix of the 120 supervectors, yielding 120 eigenvectors of dimension 2808. As K, the number of Eigenvoices that should be kept for adaptation, is not fixed, different values for K have been used to see how the results change.

The actual Eigenvoice adaptation was carried out using MLED as described in section 3.3.2, results for projection can be found in [Kuhn & Nguyen$^+$ 98]. Every Isolet speaker pronounces the alphabet twice, the first repetition was used for supervised adaptation using MLED while the second served for testing the adapted model. Results for unsupervised adaptation are published in [Kuhn & Nguyen$^+$ 00]. As both the test and the training set are recorded in the same environment the experiment does not take into account that the adaptation is usually performed in a different recording environment than the training of the SD-models [Nguyen 98, p. 46]. The adaptation was carried out with different subsets of the alphabet to see how the recognition performance changes depending on the amount of adaptation data.

In addition, an SI model was built as a baseline model from the data of the 120 training speakers. Performing recognition with the test speaker on this SI model resulted in recognition rate of 81.3%. The baseline model is represented by the horizontal dotted line in figure 7. As no speaker adaptation is applied, the system's recognition performance cannot increase if more adaptation data is available when using the SI model. Figure 7, published by Kuhn et al. in [Kuhn & Nguyen$^+$ 00], visualizes the performance of the MLED technique compared to MAP and MLLR as described in section 2.2 depending on the amount of available adaptation data. The used abbreviations are explained in table 1, samples from the underlying data are shown in table 2, where the technique with the best recognition rate for the different amounts of adaptation data is highlighted.

Figure 7 shows that for very small amounts of adaptation data - i.e. less than 15 letters -

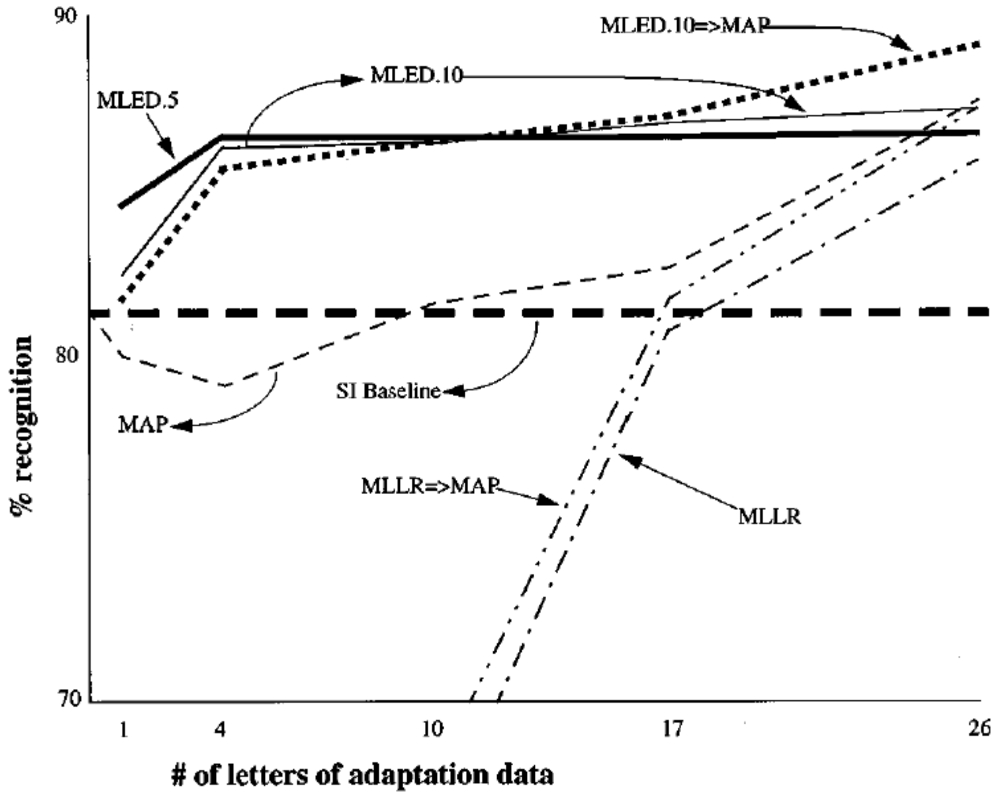

Figure 7: Performance of different adaptation methods compared to SI baseline

the recognition performance of both MAP and MLLR is not better than that of the SI approach where no adaptation at all is performed. As mentioned in section 2.2, MAP only updates the parameters of HMMs seen in adaptation so an improvement in recognition performance requires a larger amount of adaptation data. A global MLLR transform estimated from sparse data is too imprecise to increase the recognition rate. Moreover, as the letters are only spoken once in the adaptation phase, MLLR cannot estimate a reliable transformation based on one utterance [Kuhn & Nguyen$^+$ 00].

In contrast to this, the pure Eigenvoice experiment MLED.5 improves the performance of the ASR system by 3,8% relative to the SI baseline with only one letter of adaptation data. With four letters, MLED.5 and MLED.10 show equal performance, improving the SI system by 6,2% and 6,3%, respectively. The more adaptation data becomes available, the more exceeds MLED.10 the lower-dimensional MLED.5: as the 10-dimensional speaker space retains more variability between the training speakers, the Eigenvoice representation of the unknown speaker can be more precise if sufficient adaptation data is available. For larger amounts of adaptation data, the relative gains of MLED.5 and MLED.10 regarding the recognition performance become smaller. The variation covered by the low dimensional speaker spaces does not suffice to estimate a more adequate model for the unknown speaker.

The hybrid experiment MLED.10=>MAP shows that a combination of the different approaches might be a possible way to overcome their limitations. Using a model adapted with MLED.10 as a prior for MAP adaptation preserves the behaviour of Eigenvoices with sparse adaptation data and continues to improve the recognition performance if more data becomes available [Kuhn & Nguyen$^+$ 00].

Besides these experiments, Kuhn et al. tried to find interpretations for the first few Eigenvoices.

Table 1: Abbreviations used in figure 7

Index	Description
MLED.5	Eigenvoices with $K = 5$
MLED.10	Eigenvoices with $K = 10$
MAP	MAP approach with SI baseline model as prior
MLLR	global MLLR transform with a single W matrix, SI as prior
MLLR=>MAP	MAP with MLLR-adapted model as prior
MLED.10=>MAP	MAP with Eigenvoices based on $K = 10$ as prior

Table 2: Experiments with small vocabulary: Results

Adaptation technique	*Recognition result in % with different amounts of adaptation data (relative compared to SI in %)*		
	1 letter, average	{a,b,c,u}	whole alphabet
SI, no adaptation	81,3 (0,0)	81,3 (0,0)	81,3 (0,0)
Eigenvoices, K=5	**84,4 (+3,8)**	86,3 (+6,2)	86,5 (+6,4)
Eigenvoices, K=10	82,4 (+1,6)	**86,4 (+6,3)**	**87,4 (+7,5)**
MAP	80,0 (-1,6)	8,6 (-3,3)	**87,4 (+7,5)**
MLLR	3,8 (-95,3)	17,0 (-80,1)	85,8 (+5,5)

Eigenvoice one is closely related to sex (the linear combinations of all male test speakers had a positive weight for the first Eigenvoice while all but one female speakers received a negative weight). The second Eigenvoice models the amplitude of the speech, the third and forth may be related to formant movement and speech pitch, respectively [Kuhn & Nguyen$^+$ 00, p. 702–704].

3.4.2 Large Vocabulary, Continuous Speech

Experiments with speaker adaptation on large vocabulary continuous speech recognition using Eigenvoices have been carried out by Botterweck [Botterweck 00], the main results are presented below.

As mentioned in section 3.2, the main problem when applying Eigenvoice adaptation to large vocabulary recognition tasks is the little training data that is available from each training speaker, so the SD-models cannot be adequately estimated. Besides using the methods already described (building the SD-models by adapting the training speakers to a SI-model), it is crucial that the training data of every speaker covers as much of the vocabulary as possible. Botterweck used a combination of two sources for training data: the Wall Street Journal corpus with 200 speakers and about 15 minutes of speech per speaker combined with training data from an internal database that contains 50 speakers per gender with one hour of data from each speaker.

After training the initial SI-model with all 300 speakers, combined MLLR and MAP adaptation was performed with every training speaker, yielding 300 adapted models. The Eigenvoices have been derived by applying PCA to the supervectors of the models. The actual adaptation was carried out using MLED with five male and 3 female speakers that are not included in the training population.

Table 3 shows the results of the Eigenvoice technique with $K = 100$ depending on the amount of adaptation data, table 4 the effect on performance if K is varied. The values with the best

Table 3: Experiments with large vocabulary: Results depending on amount of adaptation data

Adaptation technique	*Recognition result in % with different amounts of adaptation data (relative compared to SI in %)*		
	3 seconds	12 seconds	64 seconds
SI, no adaptation	79,8 (0,0)	79,8 (0,0)	79,8 (0,0)
Eigenvoices, K=100	82,5 (+3,4)	83,8 (+5,0)	**84,0 (+5,3)**

Table 4: Experiments with large vocabulary: Results depending on dimension of speaker space

Adaptation technique	*Recognition result in % with 3 seconds of adaptation data (relative compared to SI in %)*
SI, no adaptation	79,8 (0,0)
Eigenvoices, K=1	82,3 (+3,1)
Eigenvoices, K=10	**82,8 (+3,8)**
Eigenvoices, K=20	82,5 (+3,4)
Eigenvoices, K=100	82,5 (+3,4)

relative improvement compared to a pure SI-system without any adaptation are highlighted.

The results show that for very small amounts of adaptation data, i.e. 3 seconds, there is a high relative improvement which is not much affected by the dimension of the speaker space, K. As in the small vocabulary case, the relative gains are small if more adaptation data is available.

3.5 Discussion

The main application of the Eigenvoice approach is rapid speaker adaptation: the recognition performance using a model adapted with Eigenvoices exceeds that of a SI-model even if only very sparse data is available. The results presented in section 3.4 prove that Eigenvoices outperform other approaches like MAP or MLLR in the special case of very fast adaptation.

The experimental results show that the Eigenvoice approach has crucial limitations, too. Performing excellent on sparse data on both small and large vocabulary, Eigenvoices are not able to efficiently include new information that becomes available with additional adaptation data. The approximation caused by dimensionality reduction is too imprecise if a larger amount of adaptation data is available as the precision is limited by the dimension of the speaker space. Hybrids of the techniques where Eigenvoices and MAP are combined seem to overcome this.

Besides the restricted application area regarding the amount of adaptation data, the use of Eigenvoices for large vocabulary speech recognition might cause several problems. The estimation of the speaker space can result in both numerical and storage problems, as PCA has to be applied to a parameter space of a very high dimension.

The CAT approach handles the sparse data problem in a quite similar way, yielding comparable results on fast speaker adaptation [Gales 00]. Only the method for estimating the speaker space is different between the two techniques: while Eigenvoices apply PCA, cluster adaptive training uses speaker clustering. The question if the two approaches produce a different space if they are used on the same training data is answered by Kuhn in [Kuhn & Perronnin$^+$ 01]: the "true" dimension of a space defined by K clusters might be of a lower dimension than K, while the space spanned by K Eigenvoices is always of dimension K. Figure 8 from [Kuhn & Perronnin$^+$ 01] illustrates the two-dimensional case where PCA estimates a space with $K = 2$ dimensions and CAT builds

$K = 4$ clusters. Here, CAT does not take into account that the four clusters are collinear. No experiments have been carried out yet to prove that this difference has any effect on the recognition performance.

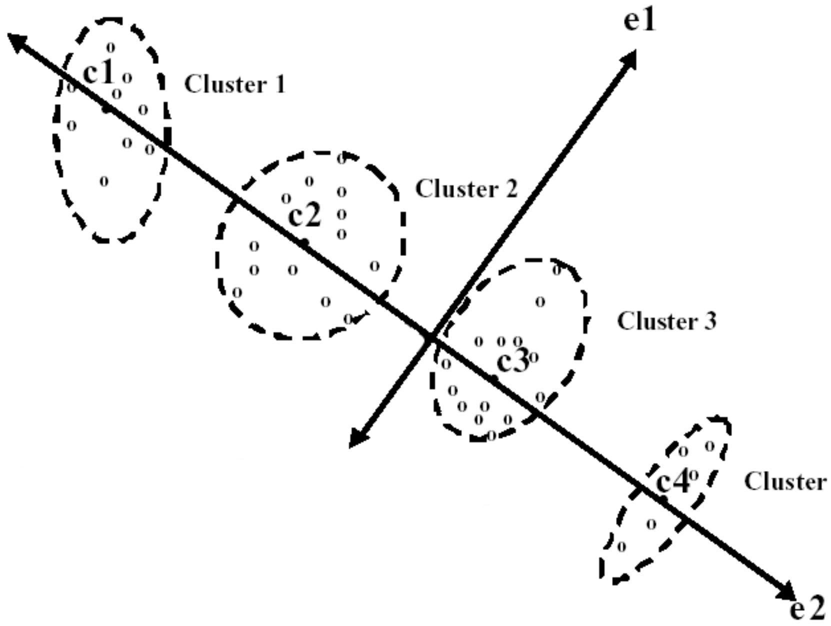

Figure 8: Eigenvoice and CAT Space

4 Conclusion

Applying speaker adaptation is a promising way to overcome the limitations of traditional speaker independent speech recognition. By using both the acoustic information from a training population and the adaptation data from an unknown speaker, the different techniques provide adapted models that can yield an increased recognition rate compared to pure SI approaches. The presentation of the different families of speaker adaptation in general as well as the description of the Eigenvoice approach in detail makes clear that the latter currently only provides a solution for a special task: rapid adaptation with very little adaptation data. Restricted to this, the approach outperforms the other techniques.

Regarding the similarity between CAT and Eigenvoices, their uniqueness has to be questioned until experimental results prove that the different spaces estimated by PCA and clustering have impact on the recognition performance.

References

[Botterweck 00] H. Botterweck: Very Fast Adaptation for Large Vocabulary Speech Recognition Using Eigenvoices. Proc. *ICSLP*, Vol. 4, pp. 354–357, Oct. 2000.

[CSLU 03] CSLU: Center for Spoken Language Understanding - The Isolet Spoken Letter Database. http://cslu.cse.ogi.edu/corpora/isolet/index.html, 2003.

[Faltlhauser 02] R. Faltlhauser: *Erzeugung robuster akustisch-phonetischer Modelle für die automatische Spracherkennung durch explizite Gruppenbildungen*. Ph.D. thesis, Technische Universität München, 2002.

[Gales 00] M.J.F. Gales: Cluster Adaptive Training of Hidden Markov Models. *IEEE Transactions on Speech and Audio Processing*, Vol. 8, No. 4, pp. 417–428, July 2000.

[Giuliani & DeMori 97] D. Giuliani, R. DeMori: *Spoken Dialogues with Computers*. Academic Press, 1997.

[Höllig & App 03] K. Höllig, A. App: Mathematik Online: Orthogonale Projektion. http://mo.mathematik.uni-stuttgart.de/kurse/kurs10/seite51.html, 2003.

[Jolliffe 86] J. Jolliffe: *Principal Component Analysis*. Springer, 1986.

[Kirby & Sirovich 90] M. Kirby, L. Sirovich: Application of the Karhunen-Loève procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, pp. 103–108, Jan. 1990.

[Kuhn & Nguyen$^+$ 98] R. Kuhn, P. Nguyen, J.C. Junqua, I. Goldwasser, N. Niedzielski, S. Fincke, K. Field, M. Contolini: Eigenvoices for Speaker Adaptation. Proc. *ICSLP*, pp. 1771–1774, 1998.

[Kuhn & Nguyen$^+$ 00] R. Kuhn, J.C.J.P. Nguyen, N. Niedzielski: Rapid Speaker Adaptation in Eigenvoice Space. *IEEE Transactions on Speech and Audio Processing*, Vol. 8, No. 6, pp. 695–707, Nov. 2000.

[Kuhn & Perronnin$^+$ 01] R. Kuhn, F. Perronnin, J. Junqua: Time is Money: Why Very Rapid Adaptation Matters. Proc. *ISCA Workshop on Adaptation Methods for Speech Recognition*, 2001.

[MIT 03] MIT: Media Laboratory Vision and Modeling Group - Photobook/Eigenfaces Demo. http://www-white.media.mit.edu/vismod/demos/facerec/system.html, 2003.

[Ney 02] H. Ney: Mustererkennung und Neuronale Netze. Lecture Notes, RWTH Aachen, 2002.

[Ney 03] H. Ney: Speech Recognition. Lecture Notes, RWTH Aachen, 2003.

[Nguyen 98] P. Nguyen: Fast Speaker Adaptation. Master's thesis, Institut Eurécom, 1998.

[Oliva 03] A. Oliva: Human and Machine Vision, Face Session 2. www.msu.edu/ aoliva/lectures/HMV_Face2_web.pdf, 2003.

[Thyes & Kuhn$^+$ 00] O. Thyes, R. Kuhn, P. Nguyen, J.C. Junqua: Speaker Identification and Verification Using Eigenvoices. Proc. *ICSLP*, Vol. 2, pp. 242–245, Oct. 2000.

[Westwood 99] R. Westwood: Speaker Adaptation Using Eigenvoices. Master's thesis, University of Cambridge, 1999.

[Woodland 01] P. Woodland: Speaker Adaptation for Continuous Density HMMs: A Review. Proc. *ISCA Workshop on Adaptation Methods for Speech Recognition*, pp. 11–19, 2001.