Environment Adaptation for Robust Speaker Verification by Cascading Maximum Likelihood Linear Regression and Reinforced Learning *

K. K. Yiu, M. W. Mak, and S. Y. Kung

October 10, 2005

Abstract

In speaker verification over public telephone networks, utterances can be obtained from different types of handsets. Different handsets may introduce different degrees of distortion to the speech signals. This paper attempts to combine a handset selector with (1) handset-specific transformations, (2) reinforced learning, and (3) stochastic feature transformation to reduce the effect caused by the acoustic distortion. Specifically, during training, the clean speaker models and background models are firstly transformed by MLLR-based handset-specific transformations using a small amount of distorted speech data. Then reinforced learning is applied to adapt the transformed models to handset-dependent speaker models and handset-dependent background models using stochastically transformed speaker patterns. During a verification session, a GMM-based handset classifier is used to identify the most likely handset used by the claimant; then the corresponding handset-dependent speaker and background model pairs are used for verification. Experimental results based on

*Paper No. CSL034-03. (Revised Version). This work was supported by the Hong Kong Polytechnic University Grant Nos. PolyU 5214/04E and PolyU 5131/02E. K. K. Yiu and M. W. Mak are with the Center for Multimedia Signal Processing, Dept. of Electronic & Information Engineering, The Hong Kong Polytechnic University. S. Y. Kung is with the Dept. of Electrical Engineering, Princeton University.

150 speakers of the HTIMIT corpus show that environment adaptation based on the combination of MLLR, reinforced learning and feature transformation outperforms CMS, Hnorm, Tnorm, and speaker model synthesis.

1 Introduction

The state-of-the-art approach to text-independent speaker verification consists in using mel-frequency cepstral coefficients (MFCCs) [1] as speaker features and Gaussian mixture models (GMMs) [2] for statistical speaker modeling. To increase the discrimination between client speakers and impostors, a GMM-based background model [3] is typically used to represent impostors' characteristics. During verification, the ratio between the likelihood that the claimant is a genuine speaker and the likelihood that the claimant is an impostor is compared against a decision threshold for decision making. In case enrollment data for individual speakers are scarce, speaker-dependent GMMs can be adapted from the background model using maximum a posteriori (MAP) techniques [3]. Because almost perfect verification has become achievable for clean and well-matched speech, researchers have focused on the problems of transducer mismatches and robustness in recent years.

Environmental robustness is an important issue in telephone-based speaker verification because users of speaker verification systems tend to use different handsets in different situations. It has been noticed that recognition accuracy degrades dramatically when users use different handsets for enrollment and verification. This lack of robustness with respect to handset variability makes speaker verification over telephone networks a challenging task.

When sufficient speech data is available from a new acoustic environment, it is sensible to retrain the speaker and background models to accommodate the new environment. However, retraining on corrupted speech requires a large amount of data from each of the possible environments. An alternative is to use speech data from different acoustic environments to train the models. This is known as multi-style training [4]. However, fine speaker characteristics will be blurred by pooling multiple training environments.

With some modifications, standard speaker adaptation methods, such as maximum a posteriori (MAP) [5] and maximum-likelihood linear regression (MLLR) [6], can be used for en-

vironment adaptation. One of the positive properties of MAP is that its performance approaches that of maximum-likelihood-based methods provided that sufficient adaptation data are available. However, MAP is an unconstrained method in that adaptation is performed only on those model parameters who have "seen" the adaptation data. MLLR, on the other hand, applies a transformation matrix to a group of acoustic centers so that all the centers are transformed. As a result, MLLR provides a quick improvement, but its performance quickly saturates as the amount of adaptation data increases.

This paper investigates two model adaptation/transformation techniques—reinforced/anti-reinforced learning of probabilistic decision-based neural networks (PDBNNs) [7] and maximum-likelihood linear regression (MLLR) [6]—in the context of telephone-based speaker verification. These techniques adapt or transform the model parameters to compensate for the *mismatch* between the training and testing conditions. We have reported in [8] some preliminary results on PDBNN and MLLR adaptation for robust speaker verification. Here, we extend the results in [8] by combining these two techniques with stochastic feature transformation (SFT) [9]. Specifically, precomputed MLLR transformation matrices are used to transform clean models to handset-dependent MLLR-adapted models. Then, PDBNN adaptation is performed on the MLLR-adapted models using handset-dependent, stochastically-transformed patterns to obtain the final adapted models. The paper also compares the proposed channel compensation methods with state-of-the-art techniques, including CMS [10], Hnorm [11], Tnorm [12], and speaker model synthesis (SMS) [13]. Experimental results based on 150 speakers of the HTIMIT corpus show that the proposed methods outperform these classical techniques.

The paper is organized as follows. In Section 2, we describe the techniques of model adaptation, including PDBNN and MLLR adaptation. Our proposed methods are detailed in Sections 2.4 and 2.5. Section 3 outlines the architecture of a handset selector that enables the verification system to select the most appropriate handset-dependent model for verification. In Section 4, speaker verification experiments that compare the effectiveness of different adaptation approaches are presented and the results are discussed in Section 5. Finally, a conclusion of the paper is provided in Section 6.

2 Model Transformation and Adaptation

To address the acoustic mismatch between training and recognition conditions, a number of compensation techniques have been proposed in the literature. These techniques can be roughly categorized into three classes: feature transformation [9], [14], [15], model transformation and adaptation [16], [13], and score normalization [11], [12]. In feature transformation, distorted speech features are transformed so that the resulting feature vectors fit the clean speaker models better. On the other hand, model transformation and adaptation modify the parameters of the statistical models so that the modified models characterize the distorted speech features better. Unlike feature transformation and model adaptation, score normalization adjusts the scores to minimize the effect introduced by handset variability. The idea is to remove the handset-dependent bias by normalizing the distributions of speaker scores using the scores of nontarget speakers. This paper focuses on model transformation and adaptation.

2.1 Speaker Model Synthesis

Recently, Teunen et al. [13] proposed a new model-based compensation method called speaker model synthesis (SMS) to address the channel mismatch problem. SMS begins with using maximum a posteriori (MAP) adaptation to adapt a channel- and gender-independent background model (also called root model) to obtain a number of channel-dependent background models, one for each known channel. The transformations among these channel-dependent background models are then computed offline. During enrollment, the channel used by the client is detected; the corresponding background model is then adapted to create a channel-dependent speaker model. During verification, the handset used by the claimant is detected. If the detected handset matches the channel of the speaker model, then the corresponding channel-dependent speaker and background models are used for verification. However, if the detected handset does not correspond to the speaker model (i.e., a mismatch occurs), then a new speaker model corresponding to the detected channel is synthesized by applying the transformation that maps the enrollment channel to the detected channel. The synthesized speaker model and the background model corresponding to the detected channel are then used for computing the log-likelihood ratio. Because all speaker and background models are adapted from the root model, a one-to-one

correspondence between individual Gaussians in these models can be preserved.

The idea of mapping speaker models from one channel to another in SMS has been extended to feature-domain channel compensation. In [15], Reynolds suggests a technique called feature mapping in which feature vectors from different channels are transformed to a channel-independent feature space by a mapping function whose parameters are learned from a set of channel-dependent models.

Feature mapping and SMS are related in that they both learn the transformations or mappings between different channels by examining the geometric differences (translation and scaling) between the corresponding channel-dependent models. While SMS focuses on synthesizing speaker models for different channels, feature mapping focuses on mapping features from different channels into a common channel-independent feature space. Both methods achieve a performance similar to Hnorm but with a fraction of the training time.

2.2 Probabilistic Decision-Based Neural Networks

Probabilistic decision-based neural networks (PDBNNs) were proposed by Lin, Kung and Lin for face detection and recognition [7], and recently PDBNNs have shown promise in text-independent speaker verification [17]. One unique feature of PDBNNs is their two-phase learning rule: locally unsupervised (LU) and globally supervised (GS). In the LU phase, PDBNNs adopt the maximum-likelihood principle to estimate the network parameters. In the globally supervised (GS) phase, discriminative training based on gradient descent and reinforced learning is applied to fine-tune the network parameters.

The following training strategy was adopted to make PDBNNs appropriate for environment adaptation. The strategy begins with the training of a clean speaker model and a clean background model using the LU training of PDBNNs. This step aims to maximize the likelihood function of the clean models given the training data. The clean models were then adapted to a handset-dependent speaker model and a handset-dependent background model using the GS training. While clean speech data were used in the LU training, distorted training data derived from the target handset were used in the GS training. The GS training uses gradient descent and reinforced learning to update the models' parameters so that the classification error of the

adapted models on the distorted data is minimized. Hence, the resulting models will be speaker- and handset-specific.

By using the distorted data derived from H handsets, the preceding training strategy produces H handset-dependent speaker models for each client speaker. Likewise, H handset-dependent background models are also created, and they are shared among all the client speakers in the system. Figure 1 (left portion) illustrates how PDBNN-based adaptation can be incorporated into a speaker verification system.

For some handsets, however, speaker-specific training data may be sparse. In such case, unsupervised adaptation such as MLLR may be more appropriate.

2.3 Maximum Likelihood Linear Regression

MLLR was originally developed for speaker adaptation [6]; however, it can also be applied to environment adaptation. Specifically, a set of adaptation matrices $W^k, k = 1, \ldots, H$, are estimated to model the mismatches between the enrollment and verification conditions; during recognition, the most appropriate transformation W^{k^*}, where $k^* \in \{1, 2, \ldots, H\}$, is applied to the Gaussian means of the speaker and background models. More precisely, if $\mu_{s,j}$ is the j-th mean vector of the clean speaker model, the adapted mean vector $\mu_{ad\_s,j}$ will be

$$\mu_{ad\_s,j} = W^{k^*} \hat{\mu}_{s,j} = A^{k^*} \mu_{s,j} + \mathbf{b}^{k^*},$$

where $\hat{\mu}_{s,j} = [{\mu_{s,j}}^T, 1]^T$ is the extended mean vector of $\mu_{s,j}$ and T denotes matrix transpose. The adaptation matrices are estimated by maximizing the likelihood of the adaptation data using the EM algorithm [18]. Each adaptation matrix is composed of a translation vector $\mathbf{b}^k \in \Re^D$ and a transformation matrix $A^k \in \Re^D \times \Re^D$, where D is the dimensionality of the feature vectors and $k = 1, \ldots, H$, i.e., $W^k = [A^k, \mathbf{b}^k]$. Figure 1 (right portion with the bypass) illustrates how MLLR-based model transformation can be incorporated into a speaker verification system.

2.4 Cascading MLLR Transformation and PDBNN Adaptation

Although PDBNN's reinforced learning uses handset-specific and speaker-specific patterns to adapt the model parameters, a previous study [8] showed that its performance is not as good as

that of MLLR adaptation. A possible reason is that PDBNN's discriminative training is sensitive to the centers' locations of the "initial model". Because the centers of the clean speaker models and the handset-dependent speaker models may be far apart, PDBNN's reinforced learning can only adapt some of the model centers in the clean speaker models. This situation is very similar to that of MAP adaptation in which adaptation is performed only on those model parameters who have "seen" the adaptation data. If a mixture component is far away from all of the adaptation data, MAP will not adapt the parameters of that mixture. To overcome this obstacle, we propose using MLLR to transform the clean speaker models to a region close to the handset-distorted speech. These MLLR-transformed models are then used as the initial models for PDBNN's reinforced learning. Figure 1 (right portion without the bypass) shows how the cascade of MLLR transformation and PDBNN adaptation can be fit into a composite speaker verification system.

2.5 Cascading MLLR Transformation and PDBNN Adaptation using Transformed Features

Although cascading MLLR transformation and PDBNN adaptation should be better than using MLLR transformation or PDBNN adaptation alone, this approach is not very practical. This is because PDBNN's reinforced learning requires client-dependent speech data to be collected from each of the environments that the client may encounter during verification. A possible solution is to use stochastic feature transformation (SFT) [9] to transform the clean speech data to environment-specific speech data. Figure 2 illustrates the idea of the proposed method.

For each known handset, a precomputed MLLR matrix is used to transform the clean models to handset-dependent MLLR-adapted models. Then, PDBNN adaptation is performed on the MLLR-adapted models using handset-dependent, stochastically-transformed patterns to obtain the final adapted models. Because handset-specific, client-dependent speech patterns are difficult to obtain, stochastic feature transformation is applied to transform the clean speaker patterns, as illustrated in Figure 2, to handset-dependent client features. The key idea is that the SFT-transformed speaker patterns are artificially generated to provide the PDBNN adaptation with the required data in the handset-distorted space for fine-tuning the MLLR-adapted models. For the background speakers' data, transformed features are not necessary because plenty of

environment-specific data are available.

2.6 A Two-Dimensional Example

Figure 3 illustrates the idea of environment adaptation in a two-class problem. Figure 3(a) plots the clean and distorted patterns of Class 1 and Class 2. The upper right (respectively lower left) clusters represent the clean (respectively distorted) patterns. The ellipses show the corresponding equal density contours. The decision boundary in Figure 3(a) was created by two 2-center GMMs that were trained using the clean patterns. As can be observed from the lower left portion of Figure 3(a), this decision boundary is not appropriate for separating the distorted patterns.

The clean models were adapted using the methods explained in Sections 2.2 through 2.4 and the results are shown in Figures 3(b) through 3(d). In Figures 3(b), 3(c), and 3(d), markers '♦' and '■' represent the centers of the clean models and '●' and '▲' represent the centers of the adapted models. The arrows indicate the adaptation of model centers and the thick curves show the equal-error decision boundaries derived from the adapted models. For PDBNN-adaptation (Figure 3(b)), two 2-center GMMs were trained independently using the clean patterns of each class. The distorted patterns of both classes were used to adapt the clean models using PDBNN's reinforced learning. For MLLR-adaptation (Figure 3(c)), a GMM was trained using the clean patterns of both classes, which was followed by the estimation of MLLR parameters using the clean GMM and the distorted patterns of both classes. The MLLR parameters were then used to transform the two clean GMMs corresponding to the two classes. For the cascaded MLLR-PDBNN adaptation (Figure 3(d)), the clean models were firstly transformed by MLLR and then PDBNN adaptation was performed on the adapted models to obtain the final models.

A comparison between Figure 3(b) and Figure 3(c) reveals that while PDBNN-adaptation can only transform some of the model centers to the region of the distorted clusters, MLLR-based adaptation can transform all of the centers of the clean GMMs to a region around the distorted clusters. By cascading MLLR transformation and PDBNN adaptation, all of the centers of the clean GMMs can be transformed to a region within the distorted clusters before the decision boundary is fine-tuned. The adaptation capability of cascading MLLR and PDBNN is also demonstrated in the speaker verification evaluation described in Section 4.

Note that although the patterns of each clean clusters were generated by a Gaussian density function, we used a 2-center GMM to model each of the two clean classes to illustrate the limitation of PDBNN adaptation. If each class is modeled by a 1-center GMM, PDBNN adaptation will always adapt the only center of the GMM. By using a GMM with more than one center, we can show that some of the centers will not be adapted by PDBNN adaptation at all.

3 Handset Selector

Unlike speaker adaptation in speech recognition where the adapted system will be used by the same speaker in subsequent sessions, in speaker verification, the claimant in each verification session may be a different person. Therefore, one cannot use the claimant's speech for adaptation, because by doing so the client's speaker model will be transformed to fit the claimant's speech regardless of the genuineness of the claimant. This will result in high false acceptance error if the claimant turns out to be an impostor. Therefore, instead of using the claimant speech for determining the transformation parameters or adapting the client model directly, we use it indirectly as follows. Before verification takes place, we obtain one set of transformation parameters (or adapted speaker models) for each of the handsets (or handset types) that the clients are likely to use. Then, during verification, we identify the most likely handset (or handset type) that is used by the claimant and select the best set of transformation parameters (or the best adapted model) accordingly.

We have adopted our recently proposed handset selector [9], [19], [20] to identify the most likely handset given an utterance. Specifically, H GMMs, $\{\Gamma_k\}_{k=1}^{H}$, as shown in Figure 1, were independently trained using the distorted speech recorded from the corresponding telephone handsets. Specifically, 100 speakers were chosen as client speakers from the HTIMIT corpus. For each handset, the utterances from the SX and SI sentence sets of other 124 background speakers were used to train the corresponding handset-dependent GMM.

During recognition, the claimant's features $\mathbf{y}(t), t = 1, \ldots, T$, were fed to all GMMs. The most likely handset k^* is selected by the Maxnet as illustrated in Figure 1. For PDBNN adaptation, the precomputed PDBNN-adapted speaker model ($\mathcal{M}_{ad\_s}^{pdbnn,k^*}$) and background model

($\mathcal{M}_{ad\_b}^{pdbnn,k^*}$) corresponding to the k^*-th handset were used for verification. For MLLR adaptation, the precomputed MLLR adaptation matrix (W^{k^*}) for the k^*-th handset was used to transform the clean speaker model ($\mathcal{M}_s$) to the MLLR-adapted speaker model ($\mathcal{M}_{ad\_s}^{mllr,k^*}$). The same matrix was also used to transform the clean background model ($\mathcal{M}_b$) to the MLLR-adapted background model ($\mathcal{M}_{ad\_b}^{mllr,k^*}$). For MLLR+PDBNN adaptation, the precomputed MLLR adaptation matrix (W^{k^*}) for the k^*-th handset was firstly used to transform the clean models ($\mathcal{M}_s$, $\mathcal{M}_b$) to the MLLR-adapted model ($\mathcal{M}_{ad\_s}^{mllr,k^*}$, $\mathcal{M}_{ad\_b}^{mllr,k^*}$). Then, PDBNN adaptation was performed on the MLLR-adapted models to obtain the MLLR+PDBNN adapted models ($\mathcal{M}_{ad\_s}^{mllr+pdbnn,k^*}$, $\mathcal{M}_{ad\_b}^{mllr+pdbnn,k^*}$). These models are used for verifying the claimant.

4 Experiments

4.1 Speech Corpus

The HTIMIT corpus [21], which contains the speech of 384 speakers, was used to evaluate the adaptation approaches. HTIMIT was collected with the objective of minimizing confounding factors so that the recorded speech differs predominantly in handset transducer effects. This is achieved by playing a gender-balanced subset of the TIMIT corpus through nine telephone handsets and a Sennheizer head-mounted microphone. Therefore, unlike other telephone speech databases where no handset labels are given, every utterance in HTIMIT is labeled with a handset name (cb1–cb4, el1–el4, pt1, or senh). Because of this special characteristic, HTIMIT has been used in several speaker recognition studies, including Quartieri et al. [22] and Mak and Kung [9].

Each speaker in the HTIMIT corpus spoke two dialectal sentences (the SA sentence set), five phonetically-compact sentences (the SX sentence set) and three phonetically-diverse sentences (the SI sentence set). Each speaker spoke the same set of sentences in the SA sentence set. In the SX sentence set, some speakers spoke the same sentences. However, all sentences in the SI sentence set are different. Therefore, the HTIMIT corpus allows us to perform text-independent speaker verification by using SA and SX sentence sets as the training set and the SI sentence set as the test set. Note that the SA sentences (i.e., the rainbow sentences) spoken by

all speakers provide the best case scenario for discriminative training techniques such as PDBNN adaptation.[1]

<table>
<tr><td colspan="3" rowspan="2"></td><td colspan="7">Adaptation / Compensation Method</td></tr>
<tr><td>CMS</td><td>Tnorm</td><td>Hnorm</td><td>SFT</td><td>MLLR</td><td>PDBNN-adaptation</td><td>SMS</td></tr>
<tr><td rowspan="8">Training</td><td rowspan="2">Creating Speaker Models</td><td>Algorithm</td><td colspan="5">EM</td><td>EM + PDBNN-adaptation</td><td>MAP</td></tr>
<tr><td>Data Used</td><td colspan="7">SA and SX sentences of the target speaker set</td></tr>
<tr><td rowspan="2">Creating Background Models</td><td>Algorithm</td><td colspan="7">EM</td></tr>
<tr><td>Data Used</td><td colspan="7">SX and SI sentences of the background speaker set</td></tr>
<tr><td rowspan="2">Creating Handset Models</td><td>Algorithm</td><td colspan="7">EM</td></tr>
<tr><td>Data Used</td><td colspan="7">SX and SI sentences of the background speaker set</td></tr>
<tr><td>Estimating Transformation Parameters</td><td>Data Used</td><td>N.A.</td><td>N.A.</td><td colspan="2">SI sentences of the antispeaker set</td><td>SA and SX sentences of the target and antispeaker sets</td><td>SX and SI sentences of the background speaker set</td></tr>
<tr><td colspan="3">Testing</td><td colspan="7">SI sentences of the impostor set</td></tr>
</table>

Table 1: Training and Testing Protocols. NA stands for not applicable. The four sets – target speaker set, background speaker set, antispeaker set, and impostor set – are mutually exclusive.

4.2 Enrollment Procedures

Speakers in the HTIMIT corpus were divided into four mutually exclusive sets: a target speaker set (consisting of 100 speakers), an antispeaker set (consisting of 100 speakers), an impostor set (consisting of 50 speakers), and a background speaker set (consisting of 124 speakers). Table 1 shows how these sets were used in the training and testing phases.

Except for SMS, each speaker in the target speaker set was assigned a 32-center GMM ($\mathcal{M}_s$) that characterizes his/her own voice.[2] The training feature vectors for each speaker model were derived from the SA and SX utterances of the corresponding speaker. A 64-center universal

[1] Readers should bear in mind that the availability of perfectly matched sentences in HTIMIT may affect the validity/portability of the results to other contexts.

[2] MAP adaptation [2] was not used because sufficient data were available for training the speaker models. Because of the intrinsic property of SMS, MAP adaptation was used for the case of SMS, resulting in 64-center speaker models.

background model ($\mathcal{M}_b$) was trained using all of the SX and SI utterances from all speakers in the background speaker set. Utterances from the head-mounted microphone (senh) were considered to be clean, and they were used for training the speaker models and the background model. The feature vectors were 12-th order mel-frequency cepstral coefficients (MFCCs) computed every 14ms using a Hamming window of 28ms.

4.3 Environment Adaptation

For PDBNN-based adaptation, the clean speaker model ($\mathcal{M}_s$) and the clean background model ($\mathcal{M}_b$) described above were used as the initial models for globally supervised training. The SA and SX utterances of the target speaker and the background speakers (excluding the target speaker) from a telephone handset were used as positive and negative training patterns, respectively. Hence, for each target speaker, a handset-dependent speaker model and a handset-dependent background model were created for each handset (including the head-mounted microphone used for enrollment).

For MLLR-based adaptation, we used a single, full transformation matrix to compensate for the mismatch between two environments. Specifically, a clean background model ($\mathcal{M}_b$) was trained using the clean speech of all speakers in the speaker set. Then, the speech data from another handset were used to estimate a transformation matrix W^k corresponding to that handset using MLLR. This procedure was repeated for all handsets in the HTIMIT corpus.

For MLLR+PDBNN adaptation, both MLLR transformation and PDBNN globally supervised training were applied to compensate for the mismatched conditions. First, MLLR was used to transform the clean speaker model ($\mathcal{M}_s$) and the clean background model ($\mathcal{M}_b$) to handset-specific models. PDBNN globally supervised training were then applied to the handset-specific models. The handset-specific SA and SX utterances of the target speaker and the antispeakers were used as the training patterns in the PDBNN supervised training.

The MLLR+PDBNN+SFT adaptation is identical to MLLR+PDBNN adaptation except that MLLR+PDBNN+SFT adaptation uses stochastically transformed speaker features in the PDBNN globally supervised training. Specifically, the clean speaker patterns from the senh handset were transformed using SFT to obtain a set of artificial, handset-specific patterns

for PDBNN adaptation. Because handset-specific speech patterns for the antispeakers can be easily obtained, it is not necessary to perform SFT on antispeakers' speech. Although HTIMIT provides training data for all speakers and handsets, not all of these data were used in this part of the experiments. This is because in practical implementation of speaker verification systems, each client speaker typically provides a few utterances from a single handset only. The experiments reported here aim to simulate a more realistic environment than HTIMIT can provide by artificially generating environment-dependent data for PDBNN adaptation. Therefore, the implementation constraints of MLLR+PDBNN adaptation—the necessity of handset-specific speaker patterns—can be relaxed.

Two sets of the SFT parameters were obtained. For the first set, the clean utterances (SI utterances from handset senh) of 20 speakers from the antispeaker set were used to create a 2-center GMM clean model Λ_X. Using this clean model and the handset-specific utterances (SI) of these 20 speakers, a set of feature transformation parameters were computed for each handset. The second set of SFT parameters was obtained similarly, except that 100 speakers from the antispeaker set were used.

A preliminary evaluation was performed to compare the performance of MLLR transformation using 5, 20 and 100 speakers to estimate the adaptation matrices W^ks. While the performance improves with the number of speakers, the computation time also increases with the total number of training patterns. However, because this is a one-time (offline) computation, the training cost is not as critical as that of the enrollment computation (see Section 5.2 for more discussions). Therefore, 20 and 100 speakers were used to estimate the MLLR transformation matrices in the experiments.

We have also conducted experiments using Hnorm [11], Tnorm [12], stochastic feature transformation (SFT) [9], and speaker model synthesis (SMS) [13] for comparison. Hnorm aims to make the scores derived from different handset types more comparable, which is an important step for ensuring consistency across handsets. Tnorm does not make explicit use of handset types; its main purpose is to correct for systematic differences in scores between test utterances. SMS is a model-based compensation approach that combines handset selector and handset-specific transformations to address the handset mismatch problem.

For Hnorm, the speech patterns derived from the handset-specific utterances of 50 same-

gender (same as the target speaker) antispeaker were used to compute the handset-dependent means and standard deviations. As a result, each target speaker model is associated with 10 handset-dependent score means and variances. These means and variances were used during verification to normalize the claimant's scores. For Tnorm [12], the verification utterances of each verification session were fed to all of the 99 nontarget speaker models to calculate the mean and variance parameters. These parameters were then used to normalize the claimant's scores. The parameters for SFT were estimated using the same 20 and 100 speakers, as in PDBNN model adaptation and MLLR model transformation. For SMS, the transformations between different handsets were derived from the parameters of the corresponding handset-dependent, gender-dependent background models, which were trained using all of the SX and SI utterances of the corresponding handsets from all gender-matched speakers in the background speaker set. Note that because of the intrinsic property of SMS, speaker models must be adapted from a universal root model. Therefore, all speakers models have 64 centers.

4.4 Verification Procedures

During verification, a pattern sequence Y derived from each of the SI sentences of the claimant was fed to the GMM-based handset selector $\{\Gamma_i\}_{i=1}^{10}$. Handset-dependent speaker and background models or adaptation matrix were selected according to the handset selector's output (see Figure 1). The test utterance was then fed to an adapted speaker model $\mathcal{M}_{ad\_s}^{\Psi,k^*}$ and an adapted background model $\mathcal{M}_{ad\_b}^{\Psi,k^*}$ to obtain the score

$$S(Y) = \log p(Y|\mathcal{M}_{ad\_s}^{\Psi,k^*}) - \log p(Y|\mathcal{M}_{ad\_b}^{\Psi,k^*}), \tag{1}$$

where $\Psi \in \{\text{PDBNN, MLLR, PDBNN+MLLR, PDBNN+MLLR+SFT}\}$ represents the type of adaptation used for obtaining the adapted models. $S(Y)$ was compared with a global, speaker-independent threshold to make a verification decision. In this work, the threshold was adjusted to determine the equal error rates (EERs).

For each handset and adaptation method, a set of client scores and a set of impostor scores were collected as follows. Each of the 100 adapted speaker models was tested by the SI utterances of the corresponding speaker and 50 impostors to produce a set of client scores and impostor

scores corresponding to that speaker.[3] For each handset in an experimental setting, there were 300 client speaker trials (100 client speakers $\times$ 3 sentences per speaker) and 15,000 impostor trials (50 impostors per client speaker $\times$ 100 client speakers $\times$ 3 sentences per impostor). The client scores and impostor scores of 100 speaker models were lumped together to form a set of client-speaker scores and a set of impostor scores. By adjusting a decision threshold, these two sets of scores produce a detection error trade-off (DET) curve [23] and a speaker-independent equal error rate (EER). Note that the DET curves and EERs were handset-dependent because they were derived from handset-dependent scores. To obtain handset-independent DET curves and EERs, the client and impostor scores of 10 handsets were concatenated to form a set of handset-independent client scores and impostor scores.

5 Results and Discussions

5.1 Comparison in Terms of Error Rates

Table 2 shows the EERs achieved by different environment adaptation approaches, including cepstral mean subtraction (CMS), test normalization (Tnorm) [12], handset normalization (Hnorm) [11], speaker model synthesis (SMS) [13], PDBNN adaptation, stochastic feature transformation (SFT) [9], MLLR, MLLR+Hnorm, MLLR+PDBNN and MLLR+PDBNN+SFT adaptation. All error rates were based on 100 target speakers and 50 impostors. The column with label "Average" shows the average of 10 handset-dependent EERs corresponding to the 10 handsets, and the column with label "Global" shows the handset-independent EERs obtained by merging the scores of 10 handsets. Therefore, the decision thresholds used for obtaining the average EERs are speaker-independent but handset-dependent, whereas the thresholds used for obtaining the global EERs are speaker- and handset-independent. Figure 4 shows the composite DET curves obtained by merging the scores of all handsets. The following discussions are based on the handset-independent EERs.

Except for Tnorm, all cases of environment adaptation show significant reduction in error rates when compared to CMS.[4] In particular, MLLR+PDBNN and MLLR+PDBNN+SFT

[3] Client speakers will not be used as impostors.

[4] The performance of Tnorm is inferior to other compensation techniques because it is not designed for cor-

Adaptation/ Compensation Method	**Equal Error Rate (%)**											
	cb1	cb2	cb3	cb4	el1	el2	el3	el4	pt1	senh	Average	Global
No compensation	16.63	21.80	33.18	32.21	14.60	26.61	14.35	25.76	29.33	2.92	**21.73**	**24.09**
CMS	10.01	9.98	23.80	18.26	10.03	12.01	12.13	9.70	11.23	8.57	**12.57**	**12.75**
Tnorm	8.88	8.94	22.58	14.94	9.30	9.78	10.40	8.64	8.51	5.54	**10.74**	**11.39**
Hnorm	9.00	8.57	15.95	12.95	9.05	10.89	10.94	9.60	11.27	8.45	**10.66**	**11.08**
SMS	6.17	6.01	16.30	11.98	6.54	7.23	8.10	6.49	8.28	5.55	**8.26**	**9.80**
PDBNN-100	8.22	8.62	9.99	10.17	7.81	11.34	7.45	10.55	9.56	3.02	**8.67**	**9.63**
SFT-20	3.63	3.95	17.91	11.57	4.69	6.70	7.25	3.92	7.64	2.97	**7.02**	**7.43**
SFT-100	3.81	3.80	18.62	11.82	4.27	6.31	6.92	3.81	7.09	3.01	**6.94**	**7.31**
MLLR-20	4.20	3.67	18.86	11.20	4.63	7.76	8.95	4.64	10.21	2.94	**7.69**	**8.34**
MLLR-100	3.70	3.55	16.40	11.01	4.34	7.10	5.90	3.65	6.60	3.01	**6.52**	**7.02**
MLLR-20+Hnorm	3.97	4.22	17.13	10.95	4.91	5.98	6.81	4.36	6.64	3.55	**6.85**	**7.49**
MLLR-100+Hnorm	4.42	4.11	16.89	10.62	4.64	5.30	5.61	4.24	5.92	3.53	**6.52**	**7.28**
SFT-20+Hnorm	4.59	3.98	14.06	10.24	4.67	7.14	5.79	4.25	6.86	3.55	**6.51**	**7.35**
SFT-100+Hnorm	4.65	3.78	13.37	10.37	4.61	6.57	5.87	4.23	7.22	3.58	**6.42**	**7.32**
MLLR+PDBNN-100	3.66	3.33	9.66	8.57	4.23	6.64	5.74	3.98	5.91	3.22	**5.49**	**6.38**
MLLR+PDBNN+SFT-100	3.67	3.32	14.29	9.45	4.20	6.93	5.66	3.98	5.85	3.22	**6.05**	**6.40**

Table 2: Equal error rates (in %) achieved by cepstral mean subtraction (CMS), Tnorm, Hnorm, speaker model synthesis (SMS), PDBNN adaptation, stochastic feature transformation (SFT), SFT+Hnorm, MLLR, MLLR+Hnorm, MLLR+PDBNN, and MLLR+PDBNN+SFT adaptation. Note that CMS and Tnorm do not require the handset selector. All results were based on 100 target speakers and 50 impostors. The MLLR and SFT transformation matrices were estimated using 20 and 100 speakers (denoted as '-20' and '-100', respectively) in the anti-speaker set. The label "Average" refers to the average of handset-dependent EERs, and the label "Global" refers to the handset-independent EERs obtained by pooling the scores of all handsets.

adaptation achieve the largest error reduction. Table 2 also demonstrates that model-based adaptation and feature-based transformation (SFT) are comparable in terms of error rate reduction. The results are also consistent with those of Figure 4, which shows that a large portion of the DET curve of SFT-100 overlaps with that of MLLR-100.

Although PDBNN adaptation uses discriminative training to adapt the model parameters to fit the new environment, the results show that using PDBNN adaptation alone is not desirable because its performance is poor than that of MLLR adaptation. This may be due to the intrinsic design of PDBNNs, i.e., the supervised reinforced/antireinforced learning of PDBNNs

recting handset differences.

are designed for fine-tuning the decision boundaries by slightly adjusting the model centers responsible for the misclassifications. As a result, some of the centers will not be adapted at all. Because only misclassified data are used for adaptation and their amount is usually small after LU training, moving all the centers from the clean space to the environmentally distorted space may be difficult. On the other hand, MLLR adaptation finds a transformation matrix to maximize the likelihood of the adaptation data. As a result, moving all of the model centers to the environmentally distorted space is much easier.

PDBNN adaptation requires speaker-specific training data from all possible handsets that the users may use. MLLR adaptation, on the other hand, only requires some environment-specific utterances to estimate the global transformation matrices, which requires much less training data; better still, the utterances do not necessarily need to be produced by the same client speaker. PDBNN adaptation also requires additional training for the inclusion of new speakers because gradient descent should be applied to adapt the clean speaker and background models to form environmental-specific models for these speakers. On the other hand, for MLLR adaptation, environment-specific transformations can be used to create new speaker models and background models once the MLLR transformation matrices have been estimated.

The results also demonstrate that SFT and MLLR achieve a comparable amount of error reduction. A comparison between SFT-20 and MLLR-20 (where the training utterances of 20 speakers were used to estimate the transformation parameters) reveals that SFT performs better when the amount of training data is small. This is because the number of free parameters in feature transformation is much less than that of MLLR. However, the performance of SFT quickly saturates when the number of training patterns increases, as indicated in SFT-100 and MLLR-100. While MLLR requires much more data to estimate the global transformation matrices robustly, its performance is better than that of SFT when sufficient training data are available.

As shown in Table 2, the EERs achieved by Tnorm and CMS are close. Note that the compensation in Tnorm and CMS are "blind" in that no handset information is utilized. The results show that Hnorm outperforms the classical CMS; however, the improvement is not as significant as SFT and MLLR. Recall that both Hnorm and Tnorm work on the likelihood-ratio score space using two normalization parameters only (bias and scale). SFT and MLLR,

on the other hand, aim to compensate for the mismatch at the feature space and model space, respectively. Both methods can translate and scale the components of feature vectors or models' centers. Because the number of free parameters in SFT and MLLR is much larger than that of Hnorm and Tnorm, SFT and MLLR are more effective provided that their free parameters can be estimated correctly.

Because SFT and Hnorm work on different spaces (SFT transforms speech features in the feature space while Hnorm performs score normalization in the score space), we have also combined these two techniques to see whether further improvement can be obtained. The results (SFT-20+Hnorm and SFT-20 in Table 2) show that combining SFT with Hnorm can improve the performance slightly. However, when a large amount of training data is available, Hnorm cannot improve performance (cf. SFT-100 and SFT-100+Hnorm in Table 2). We have also combined MLLR and Hnorm to see the effect of combining model transformation with Hnorm. Similar to the effect of Hnorm on SFT, Hnorm helps improve the performance when the amount of training data is small (cf. MLLR-20 and MLLR-20+Hnorm); but MLLR+Hnorm performs slightly worse than that of MLLR when a large amount of training data is available (cf. MLLR-100 and MLLR-100+Hnorm). A possible reason is that when a large amount of training data is available, the SFT (MLLR) has already transformed the features (clean models) to a region close to the clean features (distorted features) which does not leave much room for Hnorm to improve the performance.

Although it may not be desirable to use PDBNN's reinforced learning alone, it is amenable to the fine-tuning of the MLLR-transformed centers. This is evident from the row MLLR+PDBNN-100 in Table 2, where error reductions of 9.1% with respect to MLLR-100 and 33.7% with respect to PDBNN-100 were achieved. The main reason behind these error reductions is that MLLR can transform the clean model centers to the region of distorted clusters. These transformed centers give a better "seed" for the reinforced learning.

For MLLR+PDBNN+SFT adaptation, an EER of 6.40% is achieved (the bottom row of Table 2), which is slightly worse than that of MLLR+PDBNN adaptation (6.38%). The DET curves of MLLR+PDBNN+SFT and MLLR+PDBNN in Figure 4 are highly overlapped, which suggests that the performance of MLLR+PDBNN with and without using stochastically-transformed data are almost identical for a wide range of decision thresholds. Bear in mind

that MLLR+PDBNN adaptation requires speaker- and handset-specific training data, whereas MLLR+PDBNN+SFT adaptation requires handset-specific training data only. Therefore, in terms of practical implementation, MLLR+PDBNN+SFT adaptation has advantage over MLLR+PDBNN adaptation.

5.2 Comparison in Terms of Computational Complexity

To compare the computational complexity, we have measured the training and verification time of different adaptation approaches. The simulations were performed on a Pentium IV 2.66GHz CPU. The measurements were based on 30 target speakers, 100 background speakers, 100 antispeakers and 50 impostors in the HTIMIT corpus and the total CPU time running the whole task was recorded. The training time is composed of two parts: systemwise computation time and enrollment computation time. The systemwise computation time represents the time to carry out the tasks that only need to be performed once. These tasks include the computation of MLLR transformation matrices and background model training. The enrollment computation time represents the time to enroll a new speaker and to adapt his/her models. It was determined by taking the average enrollment time of 30 client speakers using seven utterances per speaker. The verification time is the time taken to verify a claimant based on a single utterance. It was determined by taking the average verification time of 80 claimants (30 client speakers and 50 impostors) using three utterances per claimant.

Table 3 shows the training and verification time for Hnorm, Tnorm, PDBNN-100, MLLR-100, MLLR+PDBNN-100 and MLLR+PDBNN+SFT-100. Evidently, PDBNN adaptation requires extensive computation resource during enrollment because a handset-specific speaker model and a handset-specific background model were created for each speaker and acoustic environment. It is of interest to compare the enrollment time of the techniques that involve PDBNN adaptation. Table 3 shows that the enrollment time for PDBNN-100 is longer than that of MLLR+PDBNN-100 and MLLR+PDBNN+SFT-100. Because MLLR moves the model centers to a feature space close to the adapted data, PDBNN adaptation converges rapidly. As a result, the enrollment time for MLLR+PDBNN-100 and MLLR+PDBNN+SFT-100 is less than that for PDBNN-100.

The long systemwise computation time of the methods that involve MLLR suggests that

computing the MLLR transformation matrices requires an extensive amount of computation resources. This is because to use MLLR adaptation, it is required to estimate 10 transformation matrices for 10 different environments offline, and for each environment, patterns from antispeakers are involved in the estimation of the transformation matrix. However, once the matrices have been computed, MLLR adaptation requires considerably less computation resources than PDBNN adaptation to enroll a new speaker (compare the enrollment computation time of PDBNN-100 and MLLR-100 in Table 3). This is because to enroll a speaker, PDBNN adaptation requires to create 10 speaker models and 10 background models for 10 different acoustic environments, whereas the MLLR transformation matrices have been precomputed offline. Fortunately, creating a speaker model by PDBNN adaptation involves the patterns from the corresponding speaker and the background speakers only, and adaptation takes place only for those patterns that cause misclassification.

Adaptation Method	**Training Time**		**Verification Time**
	Systemwise Computation Time (seconds)	Enrollment Computation Time (seconds)	(seconds)
Hnorm	324.30	240.78	2.48
Tnorm		0.96	37.54
PDBNN-100		8,489.54	1.20
MLLR-100	130,713.30	0.96	0.38
MLLR+PDBNN-100		2,561.74	1.59
MLLR+PDBNN+SFT-100		5,157.77	1.27

Table 3: Training and verification time used by different adaptation approaches. The training time is composed of two parts: systemwise computation time and enrollment computation time. The former represents the time to carry out the tasks that only need to be performed once. These tasks include the computation of the MLLR transformation matrices and background model training. The enrollment computation time represents the time to enroll a new speaker and to adapt his/her models. The verification time represents the time to verify a testing utterance.

Table 3 shows that Hnorm requires considerably more computation resources than Tnorm during enrollment. However, Tnorm takes a longer time to verify a speaker than Hnorm. Although both Hnorm and Tnorm work on the likelihood ratio score space, they have different complexity when it comes to training and testing. The main computation of Hnorm lies in the training phase and is proportional to the total number of nontarget utterances because these ut-

terances are fed to the speaker and background models to calculate the nontarget scores, which in turn are used to calculate the normalization parameters. Once the Hnorm parameters have been calculated, rapid verification can be achieved. For Tnorm, on the other hand, verification is computationally intensive and the amount of computation is proportional to the total number of speaker models used for deriving the normalization parameters. For a better estimation of Tnorm parameters, a large number of scores are required, which in turn requires a large number of speaker models. This can greatly lengthen the verification time. Therefore, among all adaptation methods, only Tnorm cannot achieve near real-time performance during verification.

5.3 Comparison in Terms of Storage Requirements

Because storage requirements are also important in practical implementation, it is of interest to compare the disk space usage of different adaptation methods. Table 4 shows that PDBNNs adaptation requires the largest amount of disk space because it requires the storage of one speaker model and one background model for each handset and target speaker. Hnorm, MLLR, and SFT, on the other hand, require a small amount of disk space per handset. Note that Tnorm does not require any disk storage because the normalization parameters are computed during verification.

6 Conclusions

We have presented different channel compensation approaches to addressing the problem of environmental mismatch in telephone-based speaker verification systems. These techniques can be roughly categorized into three classes: feature transformation, model transformation/adaptation and score normalization. We also combine a handset selector with (1) handset-specific transformations, (2) reinforced learning, and (3) stochastic feature transformation to reduce the effect caused by the acoustic distortion. Experimental results based on 150 speakers of the HTIMIT corpus show that the model adaptation based on the combination of MLLR and PDBNN outperforms a number of classical techniques, including CMS, stochastic feature transformation, Hnorm, and speaker model synthesis.

Adaptation Method	Number of Parameters	Disk Usage in Bytes
CMS (No adaptation)	$PN_c + N_b$	326,400
Hnorm	$PN_c + N_b + 2PE$	334,400
SMS	$PN_c + 2EN_b$	768,000
MLLR	$PN_c + N_b + D(D+1)E$	332,640
SFT	$PN_c + N_b + 2DE$	327,360
PDBNN	$(PN_c + N_b)E$	3,264,000

P = Population size (100)

D = Feature dimension (12)

N_c = Number of parameters per speaker model ($= [1 + D + D]32 = 800$)

N_b = Number of parameters per background model ($= [1 + D + D]64 = 1{,}600$)

E = number of acoustic environments (10)

Table 4: Disk space requirements of different adaptation methods. The numbers inside brackets are the values of the corresponding parameters used in the experiments. *Note:* The figures in the last column are calculated by assuming that each floating-point number occupies 4 bytes. For SMS, $N_c = N_b = 1{,}600$ because speaker models must be adapted from background models.

References

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

[2] D. A. Reynolds and R. C. Rose. Robust text-independent speaker identification using Gaussian mixture speaker models. *IEEE Trans. on Speech and Audio Processing*, 3(1):72–83, 1995.

[3] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10:19–41, 2000.

[4] R. Lippmann, E. Martin, and D. Paul. Multi-style training for robust isolated-word speech recognition. In *Proc. ICASSP'87*, pages 705–708, 1987.

[5] C. H. Lee, C. H. Lin, and B. H. Juang. A study on speaker adaptation of the parameters of continuous density hidden Markov models. *IEEE Trans. on Acoustic, Speech, and Signal Processing*, 39(4):806–814, April 1991.

[6] C. J. Leggetter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9(4):806–814, 1995.

[7] S. H. Lin, S. Y. Kung, and L. J. Lin. Face recognition/detection by probabilistic decision-based neural network. *IEEE Trans. on Neural Networks, Special Issue on Biometric Identification*, 8(1):114–132, 1997.

[8] K. K. Yiu, M. W. Mak, and S. Y. Kung. Environment adaptation for robust speaker verification. In *Eurospeech'03*, pages 2973–2976, 2003.

[9] M. W. Mak and S. Y. Kung. Combining stochastic feature transformation and handset identification for telephone-based speaker verification. In *Proc. ICASSP'02*, pages I701–I704, 2002.

[10] B. S. Atal. Effectiveness of linear prediction characteristics of the speech wave for automatic speaker for automatic speaker identification and verification. *J. Acoust. Soc. Am.*, 55(6):1304–1312, 1974.

[11] D. A. Reynolds. Comparison of background normalization methods for text independent speaker verification. In *Eurospeech'97*, pages 963–966, 1997.

[12] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas. Score normalization for text-independent speaker verification systems. *Digital Signal Processing*, 10:42–54, 2000.

[13] R. Teunen, B. Shahshahani, and L. Heck. A model-based transformational approach to robust speaker recognition. In *ICSLP 2000*, volume 2, pages 495–498, 2000.

[14] A. Sankar and C. H. Lee. A maximum-likelihood approach to stochastic matching for robust speech recognition. *IEEE Trans. on Speech and Audio Processing*, 4(3):190–202, 1996.

[15] D. A. Reynolds. Channel robust speaker verification via feature mapping. In *ICASSP-03*, volume 2, pages 53–56, 2003.

[16] F. Beaufays and M. Weintraub. Model transformation for robust speaker recognition from telephone data. In *ICASSP-97*, volume 2, pages 1063–1066, 1997.

[17] K. K. Yiu, M. W. Mak, and S. Y. Kung. A comparative study on kernel-based probabilistic neural networks for speaker verification. *International Journal of Neural Systems*, 12(5):381–391, 2002.

[18] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. of Royal Statistical Soc., Ser. B.*, 39(1):1–38, 1977.

[19] C. L. Tsang, M. W. Mak, and S. Y. Kung. Divergence-based out-of-class rejection for telephone handset identification. In *Proc. Int. Conf. on Spoken Language Processing*, pages 2329–2332, 2002.

[20] M. W. Mak, C. L. Tsang, and S. Y. Kung. Stochastic feature transformation with divergence-based out-of-handset rejection for robust speaker verification. *EURASIP J. on Applied Signal Processing*, 4:452–465, 2004.

[21] D. A. Reynolds. HTIMIT and LLHDB: speech corpora for the study of handset transducer effects. In *ICASSP'97*, volume 2, pages 1535–1538, 1997.

[22] T. F. Quartieri, D. A. Reynolds, and G. C. O'Leary. Estimation of handset nonlinearity with application to speaker recognition. *IEEE Trans. on Speech and Audio Processing*, 8(5):567–584, 2000.

[23] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET Curve in assessment of detection task performance. In *Eurospeech'97*, pages 1895–1898, 1997.

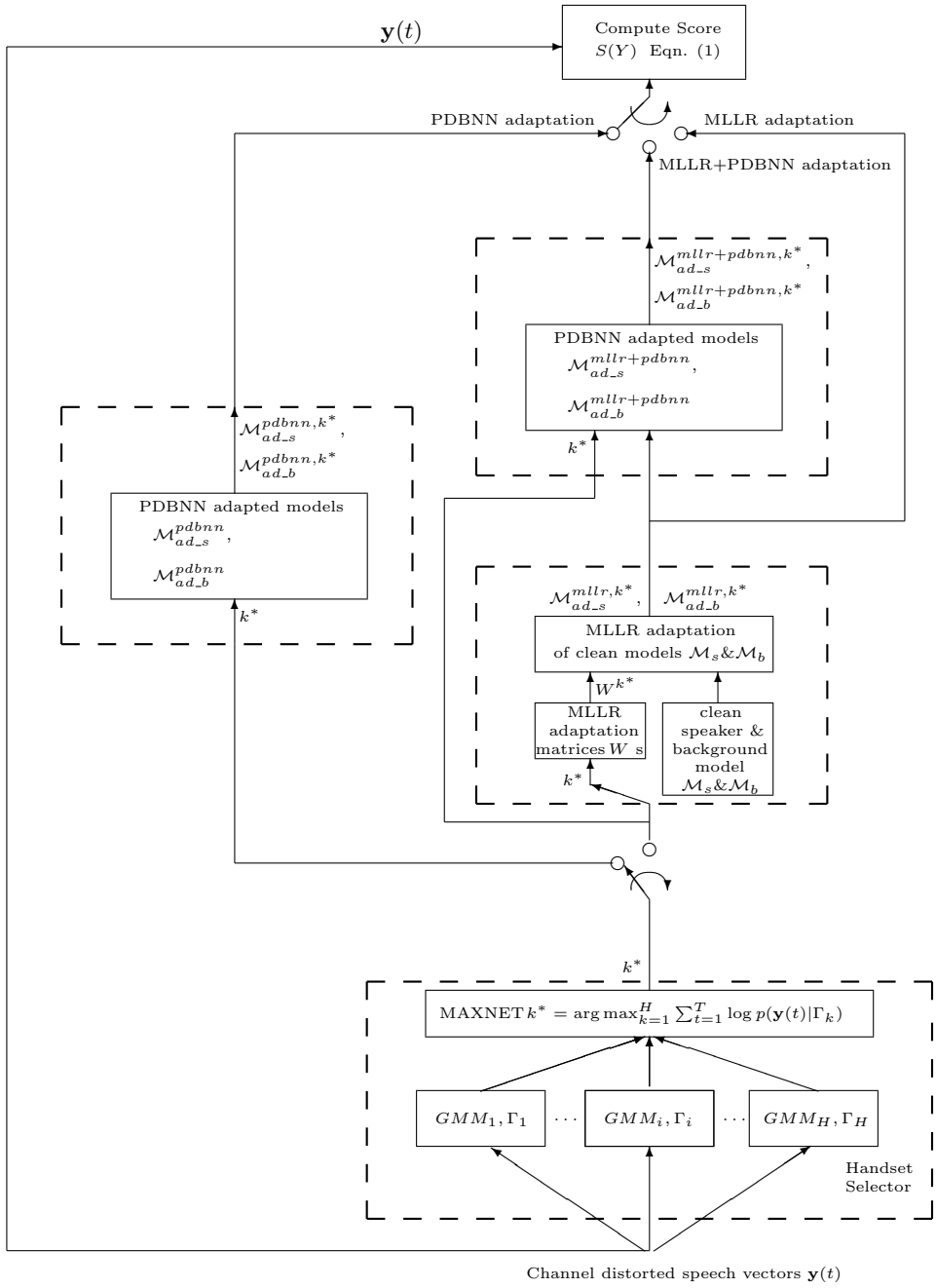


Figure 1: The combination of handset identification and model adaptation for robust speaker verification. *Note:* adaptation was applied to both the speaker models and background models.

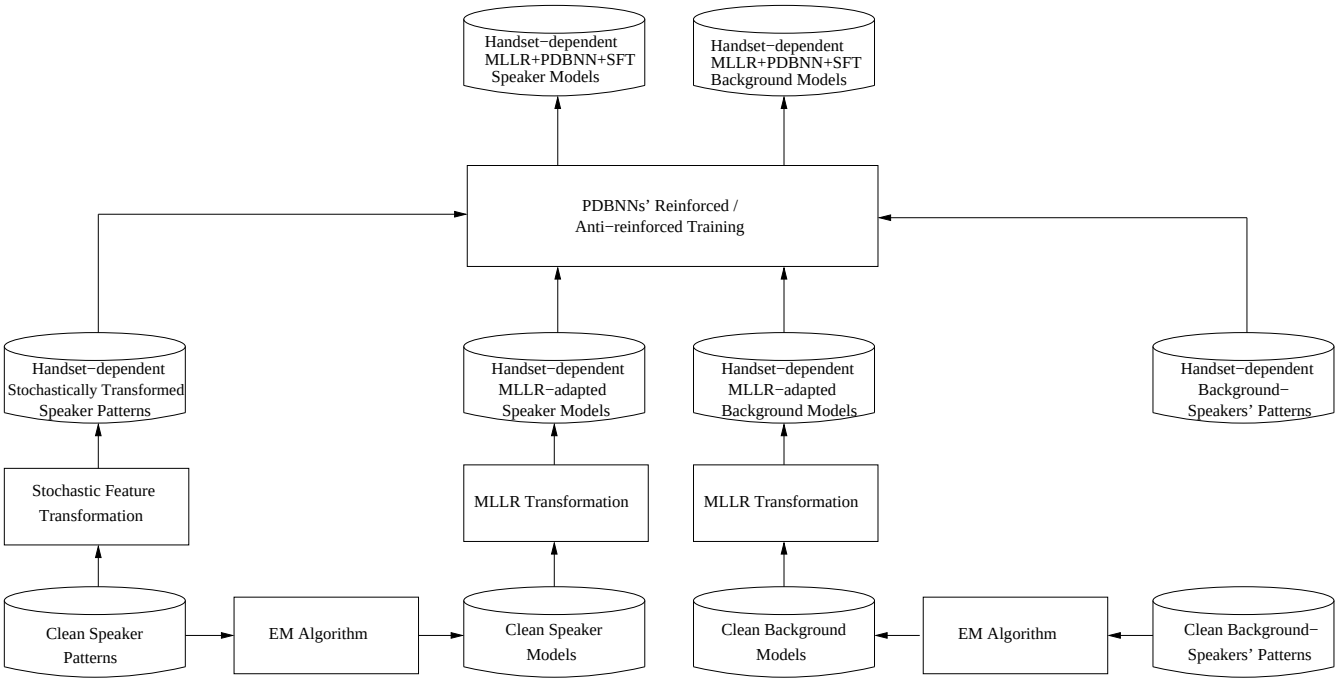
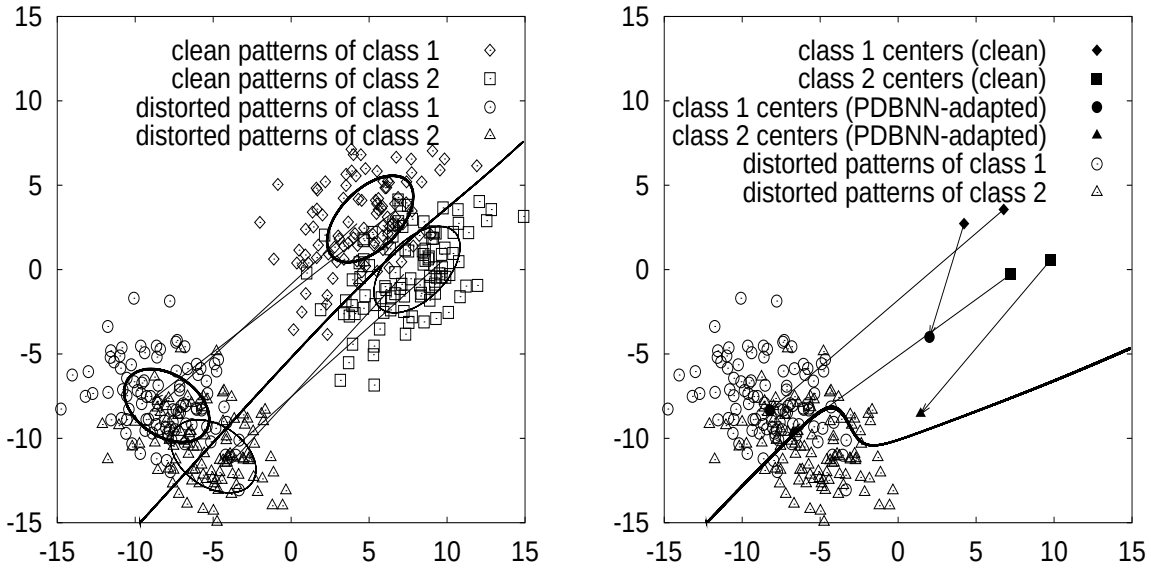
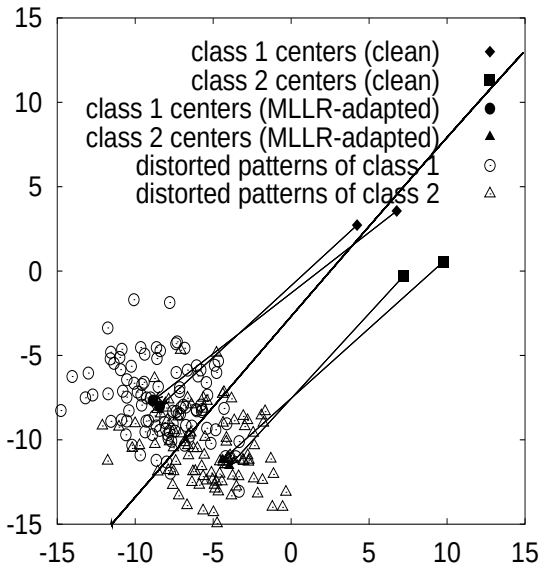


Figure 2: The process of fine-tunning MLLR-adapted models using transformed features. For each known handset, a precomputed MLLR adaptation matrix is used to transform the clean models to handset-dependent MLLR-adapted models. Then, PDBNN adaptation is performed on the MLLR-adapted models using handset-dependent, stochastically-transformed patterns to obtain the final adapted models.

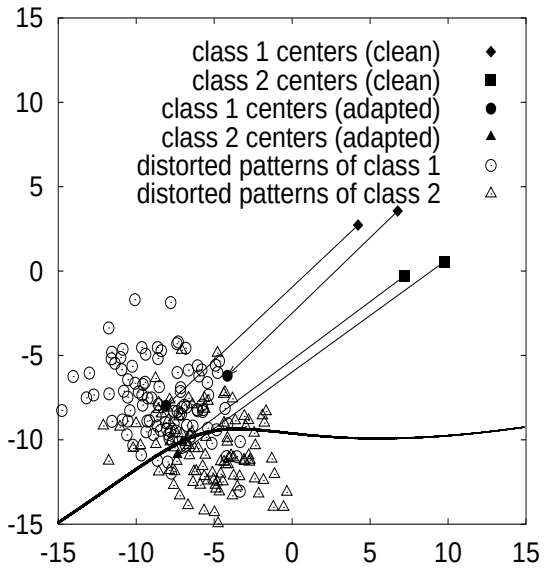


(a) 2-class problem

(b) PDBNN adaptation



(c) MLLR adaptation



(d) MLLR+PDBNN adaptation

Figure 3: (a) Scatter plots of the clean and distorted patterns in a 2-class problem. The thick and thin ellipses represent the equal density contour of Class 1 and Class 2, respectively. The upper right (respectively lower left) clusters contain the clean (respectively distorted) patterns. The decision boundary (thick curve) was derived from the clean GMMs which were trained using the clean patterns. (b) Centers of the clean models and the PDBNN-adapted models. The thick curve is the decision boundary created by the PDBNN-adapted models. (c) Centers of the clean models and MLLR-adapted models. The thick curve is the decision boundary created by the MLLR-adapted models. (d) Centers of the clean models and MLLR+PDBNN adapted models. (The clean models were firstly transformed by MLLR; then the transformed models were further adapted by PDBNN adaptation to obtain the final model.) In (b), (c), and (d), only the distorted patterns are plotted for clarity. The arrows indicate the displacement of the original centers and the adapted centers.

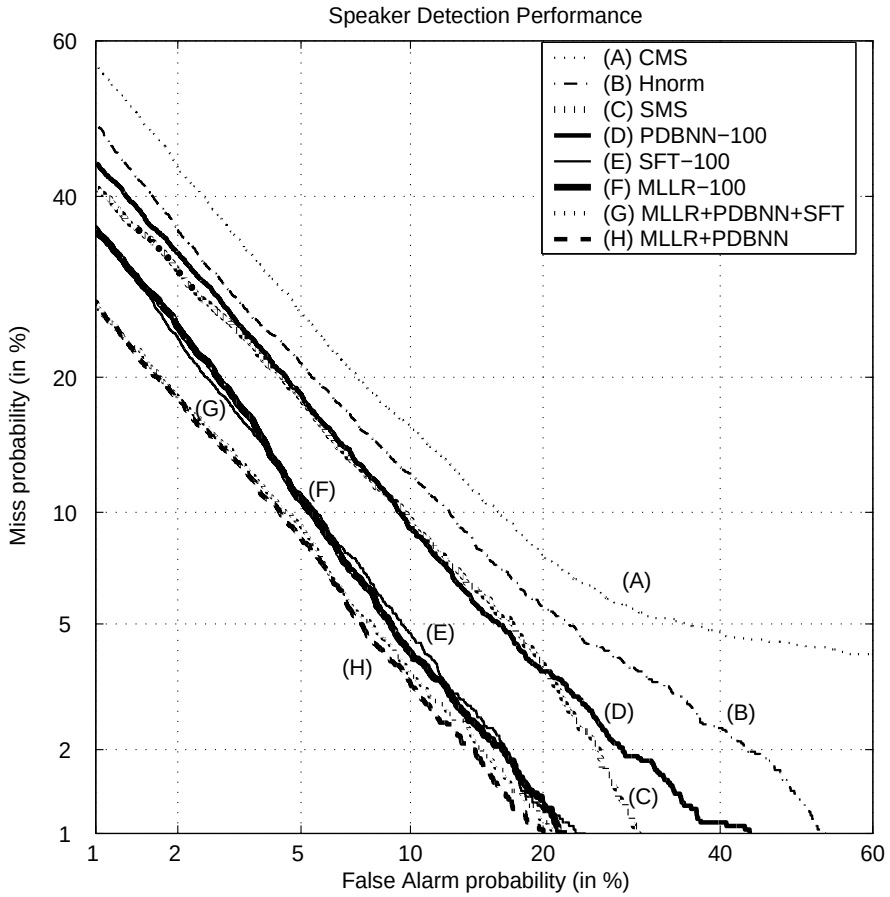


Figure 4: Composite DET curves based on different environment adaptation approaches: cepstral mean subtraction (CMS), handset normalization (Hnorm), speaker model synthesis (SMS), PDBNN, stochastic feature transformation (SFT), MLLR, MLLR+PDBNN+SFT, and MLLR+PDBNN. The composite DET curve is created by merging the scores of all speakers and handsets. For ease of comparison, methods labelled from (A) to (H) in the legend are arranged in descending order of EERs.