An Environment-Compensated Minimum Classification Error Training Approach Based on Stochastic Vector Mapping

Jian Wu, *Member, IEEE*, and Qiang Huo, *Member, IEEE*

***Abstract*—A conventional feature compensation module for robust automatic speech recognition is usually designed separately from the training of hidden Markov model (HMM) parameters of the recognizer, albeit a maximum-likelihood (ML) criterion might be used in both designs. In this paper, we present an environment-compensated minimum classification error (MCE) training approach for the joint design of the feature compensation module and the recognizer itself. The feature compensation module is based on a stochastic vector mapping function whose parameters have to be learned from stereo data in a previous approach called SPLICE. In our proposed MCE joint design approach, by initializing the parameters with an approximate ML training procedure, the requirement of stereo data can be removed. By evaluating the proposed approach on Aurora2 connected digits database, a digit recognition error rate, averaged on all three test sets, of 5.66% is achieved for multicondition training. In comparison with the performance achieved by the baseline system using ETSI advanced front-end, our approach achieves an additional overall error rate reduction of 12.4%.**

***Index Terms*—Feature compensation, hidden Markov model (HMM), minimum classification error training (MCE), noise robustness, robust speech recognition, stochastic vector mapping.**

I. Introduction

IT IS WELL known that the performance of an automatic speech recognition (ASR) system will be deteriorated in mismatched training and test conditions. In the past two decades, there were many efforts proposed to alleviate such degradation caused by the additive noise and convolutional distortion. However, the performance achieved by most of them are unable to reach that under matched training and test conditions. Recently, Microsoft researchers [9], [10] demonstrated that this performance limit could be surpassed by using a feature compensation technique called Stereo-based piecewise linear compensation for environments (SPLICE) and a noise adaptive training (NAT) strategy. SPLICE is an extension of the feature compensation techniques developed at Carnegie Mellon University (CMU) in the past decade (e.g., [1], [26], [31], [32]). NAT is a combination of ideas of multistyle training [23] and irrelevant variability normalization (e.g., the simple practice of performing cepstral mean normalization in both training and testing, and other more complex techniques described in, e.g., [2], [7], [14], [15], [17], [18], [25], [34], [35]). As a further exploration, removing the requirement of stereo-data in SPLICE yet achieving a high performance is the primary motivation of the work reported in this paper.

For SPLICE and many other *frame-dependent* bias removal algorithms (e.g., [1], [21], [26], [29], [31] and references therein), in both training and testing, "corrupted" speech features are mapped into "clean" speech features by a simple transformation, which is referred to as a *stochastic vector mapping* (SVM) in this paper. After each training utterance is enhanced, a multistyle training is performed on all of the "pseudoclean" feature vectors to estimate the hidden Markov model (HMM) parameters of a speech recognizer. Apparently, the success of such a framework relies on at least the correctness of the following assumptions.

- The mismatch between clean and noisy data in feature domain can be compensated by the assumed stochastic vector mapping.
- The residue distortion after feature compensation can be modeled and absorbed by the collectively trained HMMs.

However, two sets of parameters, namely, the parameters of feature compensation module and the HMM parameters of the recognizer, are typically estimated separately using a maximum-likelihood (ML) criterion. This cannot guarantee to achieve the objective of minimum classification error (MCE) in recognition. Historically, a pioneering work on discriminative feature extraction (DFE) using MCE criterion [19] was first reported in [3]. Since then, it has been demonstrated by many researchers in various speech applications that MCE criterion can be beneficially applied to the design of a feature extractor either separately from or jointly with the training of recognizer parameters (e.g., [4], [5], [24], [27], [28], [33], [39]). In [7], a signal-conditioned MCE training approach was proposed and demonstrated to be effective for compensating distortions caused by both the channel mismatch as well as additive noises [6], [7]. Although the HMM parameters are MCE-trained on the compensated feature vectors, the feature compensation module is derived directly under another criterion [29] from the updated HMM parameters during their MCE training. It is thus well-motivated to use an MCE criterion for the *joint* design of the above two sets of parameters. This explains the second motivation of the work reported in this paper.

Manuscript received September 30, 2004; revised October 14, 2005. This work was supported by the RGC of the Hong Kong SAR under Project Numbers HKU7022/00E and HKU7039/02E. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Li Deng.

J. Wu was with the Department of Computer Science, University of Hong Kong, Hong Kong, China. He is now with the Microsoft Corporation, Redmond, WA 98052-6399 USA (e-mail: jianwu@microsoft.com).

Q. Huo is with the Department of Computer Science, University of Hong Kong, Hong Kong, China (e-mail: qhuo@cs.hku.hk).

Digital Object Identifier 10.1109/TASL.2006.872616

Inspired by the performance potential demonstrated in "SPLICE + NAT" approach and the past success of MCE-based DFE approach, we proposed in [36] an environment compensated MCE training approach for the *joint* design of the feature compensation module and the recognizer itself. The effectiveness of the proposed approach has been confirmed ([36], [37]) in a series of benchmark test on Aurora2 connected digits database [16]. In this paper, we summarize our previous work on this topic with additional experimental results and hopefully make it more accessible to the general readership.

The rest of the paper is organized as follows. In Section II, we review the framework of the conventional stochastic vector mapping approach using stereo data for environment compensation and establish the necessary notations. In Section III, we describe the proposed approach of joint design where stereo data are not required. In Section IV, we report the evaluation results on Aurora2 database to demonstrate the effectiveness of the proposed approach. Finally, we conclude the paper in Section V.

II. ML-Based Estimation of Stochastic Vector Mapping Function Parameters From Stereo Data and HMM Design for Environmental Compensation

Let us assume that a speech utterance corrupted by additive noise and convolutional distortion is transformed by signal processing operations into a sequence of feature vectors. Then, the task of *frame-dependent* feature-based environmental compensation is to estimate the clean speech feature vector $\hat{x}$ From the noisy speech feature vector y, by applying the environment dependent transformation $\mathcal{F}(y;\Theta^{(e_y)})$, where Θ represents the trainable parameters associated with the transformation and e_y denotes the corresponding environment class (e.g., a combination of noise type and noise level) to which y belongs. For the simplicity of notation, the subscript y in e_y will be ignored hereinafter if no confusion arises. Obviously, $\mathcal{F}(\cdot)$ is a highly nonlinear function of y that is difficult to characterize analytically. One of the feasible solutions used in most of successful feature mapping approaches such as CMUs algorithms and SPLICE, is to approximate it by using a stochastic vector mapping approach as described in the following.

A. Estimating Correction Vectors Using Stereo Data: SPLICE Approach

Given a set of training data $\mathcal{Y} = \{Y_i\}_{i=1}^{I}$, where $Y_i = \{y_{i1},\cdots,y_{iT_i}\}$ is a sequence of feature vectors of noisy speech, suppose that their distortions can be partitioned into E classes of background environments. Assume that the D-dimensional feature vector y under an environment class e follows the distribution of a mixture of Gaussian probability density functions (PDF)

$$p(y|e)=\sum_{k=1}^{K}p(k|e)p(y|k,e)=\sum_{k=1}^{K}p(k|e)\mathcal{N}\left(y;\xi_k^{(e)},R_k^{(e)}\right) \tag{1}$$

where $\mathcal{N}(\cdot;\xi,R)$ is a normal distribution with mean vector ξ and diagonal covariance matrix R. These model parameters can be estimated easily from the corresponding set of training data with EM algorithm [8]. Based on the above notations, in the stochastic vector mapping approach like [36], the mapping function can be defined as follows [10]:

$$\hat{x}\triangleq\mathcal{F}\left(y;\Theta^{(e)}\right)=y+\sum_{k=1}^{K}p(k|y,e)b_k^{(e)} \tag{2}$$

where

$$p(k|y,e)=\frac{p(k|e)p(y|k,e)}{\sum_{j=1}^{K}p(j|e)p(y|j,e)} \tag{3}$$

and $\Theta^{(e)}=\{b_k^{(e)}\}_{k=1}^{K}$ is the set of mapping function parameters (also referred to as *correction vectors* hereinafter) that can be estimated from the training data by using an ML criterion.

If stereo data for both clean and noisy speech are available and used to estimate $\Theta^{(e)}$, this becomes the SPLICE approach [10], which is a modification and extension of the fixed codeword-dependent cepstral normalization (FCDCN) algorithm described in [1]. For example, suppose that $\{(x_{it},y_{it})\}_e$ denotes the set of corresponding pairs of clean and noisy speech feature vectors recorded under a particular environment class e. Let us consider a particular environment class e and use I_e to denote the set of subscripts of training utterances belonging to the environment class e. Then, the correction vectors can be trained based on an ML criterion as follows:

$$b_k^{(e)}=\frac{\sum_{i\in I_e}\sum_t p(k|y_{it},e)(x_{it}-y_{it})}{\sum_{i\in I_e}\sum_t p(k|y_{it},e)}. \tag{4}$$

B. Training CDHMMs With Pseudoclean Speech

As assumed previously, each training environment class e can be characterized by a Gaussian mixture model (GMM) as shown in (1) and a stochastic vector mapping function as shown in (2). If we transform each training feature vector accordingly, a pseudoclean training set, $\hat{\mathcal{X}}=\{\hat{X}_i\}_{i=1}^{I}$, can be created. Further suppose that in our speech recognizer, each basic speech unit is modeled by a Gaussian mixture continuous-density HMM (CDHMM), whose parameters are denoted as $\lambda=\{\pi_s,a_{ss'},c_{sm},\mu_{sm},\Sigma_{sm};s,s'=1,\cdots,S;m=1,\cdots,M\}$, where S is the number of states, M is the number of Gaussian components for each state, $\{\pi_s\}$ is the initial state distribution, $a_{ss'}$'s are state transition probabilities, c_{sm}'s are Gaussian mixture weights, $\mu_{sm}=[\mu_{sm1},\cdots,\mu_{smD}]^{Tr}$ is a D-dimensional mean vector, and $\Sigma_{sm}=diag\{\sigma_{sm1}^2,\cdots,\sigma_{smD}^2\}$ is a diagonal covariance matrix. Consequently, the set of pseudoclean HMM parameters in our speech recognizer, $\Lambda=\{\lambda\}$, can be estimated by using an ML- or MCE-based training strategy.

C. Recognition Process Using Stochastic Vector Mapping

In recognition, given an unknown utterance Y_i, the most similar training environment class is first identified as that of having the maximum-likelihood $p(Y_i|e)=\prod_{t=1}^{T_i}p(y_{it}|e)$ for $e=1,2,\cdots,E$. Then, the corresponding GMM and the mapping function are used to derive a pseudoclean version of

$\hat{X}_i$ from Y_i. For the convenience of notation, we also use hereinafter $\mathcal{F}(Y_i;\Theta)$ to denote the enhanced version of the utterance Y_i by transforming individual feature vector y_{it} as defined in (2). After feature compensation, $\hat{X}_i$ is finally recognized by a pseudoclean CDHMM-based recognizer trained as described.

III. Joint Design of Stochastic Vector Mapping Function and CDHMMs Without Using Stereo Data

In many application scenarios, stereo data are too expensive to collect, thus not available for estimating vector mapping function parameters. In this section, an environment-compensated MCE training approach is proposed for the joint design of the mapping function parameters and CDHMM parameters that does not rely on the availability of stereo data.

A. ML-Based Joint Design Approach

In order to provide reasonable initial values for MCE training, an approximate ML training procedure is developed to estimate the vector mapping function parameters Θ and CDHMM parameters Λ and is described as follows.

1) Initialization: First, a set of CDHMMs Λ are trained from multicondition training data and used as the initial values of HMM parameters. Initial values of the correction vectors $\{b_k^{(e)} = [b_{k1}^{(e)},\cdots,b_{kD}^{(e)}]^{Tr}\}$ are set to be zero.

2) Estimating Vector Mapping Function Parameters: Second, for each environment class e, one EM iteration is performed to estimate the environment dependent mapping function parameters $\bar{\Theta}^{(e)}$ to increase the log-likelihood function $\sum_{i=1}^{I}\log p(\mathcal{F}(Y_i;\Theta)|\Lambda)$.

By using the general EM algorithm [8] and the specific mapping function in (2), the auxiliary $\mathcal{Q}$-function for $\Theta^{(e)}$ can be derived as

$$\begin{aligned}\mathcal{Q}_e &= \sum_{i\in I_e}\sum_t\sum_s\sum_m \zeta_{it}(s,m)\\ &\quad \log\mathcal{N}\left(y_{it}+\sum_k p(k|y_{it},e)b_k^{(e)};\mu_{sm},\Sigma_{sm}\right)\\ &= \sum_{i\in I_e}\sum_t\sum_s\sum_m \zeta_{it}(s,m)\\ &\quad\times\left[y_{it}+\sum_k p(k|y_{it},e)b_k^{(e)}-\mu_{sm}\right]^{Tr}\Sigma_{sm}^{-1}\\ &\quad\left[y_{it}+\sum_k p(k|y_{it},e)b_k^{(e)}-\mu_{sm}\right]+\text{Const.}\end{aligned}\tag{5}$$

In the previous equation, $\zeta_{it}(s,m)$ is the occupation probability of Gaussian component m in state s, at time t of current observation. It can be calculated with a Forward-Backward procedure using training utterance $\hat{X}_i$ (enhanced from Y_i with current Θ) against current HMM parameters Λ in the E-step. $y_{it} = [y_{it1},\cdots,y_{itD}]^{Tr}$ is the feature vector of utterance Y_i at tth frame. $Const$ is a term irrelevant to $b_k^{(e)}$.

By setting the derivative of $\mathcal{Q}_e$ with respect to $b_k^{(e)}$ as zero, we have

$$\begin{aligned}&\sum_{i\in I_e}\sum_t\sum_s\sum_m\sum_{k'}\zeta_{it}(s,m)p(k|y_{it},e)p(k'|y_{it},e)\Sigma_{sm}^{-1}b_{k'}^{(e)}\\ &=\sum_{i\in I_e}\sum_t\sum_s\sum_m\zeta_{it}(s,m)p(k|y_{it},e)\Sigma_{sm}^{-1}(\mu_{sm}-y_{it}).\end{aligned}\tag{6}$$

Since above equation holds for all k, it is equivalent to solve the root of vector $\mathcal{B}_d^{(e)} = [b_{1d}^{(e)},\cdots,b_{Kd}^{(e)}]^{Tr}$ in the following equation:

$$\mathcal{A}_d^{(e)}\mathcal{B}_d^{(e)} = \mathcal{C}_d^{(e)}\tag{7}$$

where $\mathcal{A}_d^{(e)}$ is a $K\times K$ matrix with the (k,k')th element being

$$\mathrm{a}_d^{(e)}(k,k') = \sum_{i\in I_e}\sum_t\left[\sum_s\sum_m\frac{\zeta_{it}(s,m)}{\sigma_{smd}^2}\right]p(k|y_{it},e)p(k'|y_{it},e)\tag{8}$$

and $\mathcal{C}_d^{(e)}$ is a K-dimensional vector $[c_d^{(e)}(1),\cdots,c_d^{(e)}(K)]^{Tr}$ with

$$c_d^{(e)}(k) = \sum_{i\in I_e}\sum_t\left[\sum_s\sum_m\frac{\zeta_{it}(s,m)(\mu_{smd}-y_{itd})}{\sigma_{smd}^2}\right]p(k|y_{it},e)\tag{9}$$

for all $k=1,\cdots,K$. Therefore, the estimation of $b_k^{(e)}$ involves the calculation of solving D equations like (7), of which each needs an inverse operation of the $K\times K$ matrix.

To reduce computational efforts for estimating $\Theta^{(e)}$ as described above, we can instead adopt a vector mapping function simpler than (2) as

$$\hat{x} = y + b_k^{(e)}\tag{10}$$

where $k = \arg\max_{k'=1,\cdots,K} p(k'|y,e)$ for the environment class e that y belongs to [9]. Then, it can be simply derived as

$$b_{kd}^{(e)} = \frac{\sum_{i\in I_e}\sum_t\sum_s\sum_m\frac{\mathbf{1}\left[k=\arg\max_{k'}p(k'|y_{it},e)\right]\zeta_{it}(s,m)(\mu_{smd}-y_{itd})}{\sigma_{smd}^2}}{\sum_{i\in I_e}\sum_t\sum_s\sum_m\frac{\mathbf{1}[k=\arg\max_{k'}p(k'|y_{it},e)]\zeta_{it}(s,m)}{\sigma_{smd}^2}}\tag{11}$$

where

$$\mathbf{1}(a=b) = \begin{cases}1, & \text{if } a \text{ equals to } b\\ 0, & \text{otherwise.}\end{cases}\tag{12}$$

This is the approach we used in our experiments for this study. It is noted that the above updating formula is similar to the bias estimation formulas of feature-space stochastic matching approach reported in [30].

3) Estimating CDHMM Parameters: Third, we transform each training utterance using (2) with the relevant mapping function parameters $\bar{\Theta}^{(e)}$. Using the resulted pseudoclean utterances, one EM iteration is performed to reestimate CDHMM parameters $\bar{\Lambda}$, with an increase of the likelihood function $\sum_{i=1}^{I}\log p(\mathcal{F}(Y_i;\bar{\Theta})|\Lambda)$.

After the above three steps, we obtain the $\bar{\Theta}$ and $\bar{\Lambda}$ as an approximate ML estimation of mapping function parameters and CDHMM parameters, which are used for recognition directly or as initial values for further MCE training.

B. MCE-Based Joint Design Approach

In the proposed environment compensated MCE training approach, it still uses the stochastic vector mapping function as defined in (2). However, the mapping function parameters Θ and the CDHMM parameters Λ are estimated jointly by minimizing the following empirical classification error defined on the training data $\mathcal{Y} = \{Y_i\}_{i=1}^{I}$

$$\ell(\Theta, \Lambda) = \frac{1}{I}\sum_{i=1}^{I} l\left(\mathcal{F}(Y_i;\Theta);\Lambda\right) \tag{13}$$

with $l(\mathcal{F}(Y;\Theta);\Lambda)$ being the loss function for the training utterance Y defined as follows:

$$l\left(\mathcal{F}(Y;\Theta);\Lambda\right) = \frac{1}{1+exp\left(-\alpha d\left(\mathcal{F}(Y;\Theta);\Lambda\right)+\beta\right)} \tag{14}$$

where α and β are two control parameters (e.g., [20]). In the previous equation, $d(\cdot)$ is a misclassification measure defined as

$$d\left(\mathcal{F}(Y;\Theta);\Lambda\right) = -g\left(\mathcal{F}(Y;\Theta);\Lambda\right) + \bar{g}\left(\mathcal{F}(Y;\Theta);\Lambda\right) \tag{15}$$

where $g(\cdot)$ is a discriminant function for recognition decision-making, and $\bar{g}(\cdot)$ is an antidiscriminant function. The form of these two functions depends on the definition of the "class" in the context of MCE. In our experiments, the entire word string is referred to as a "class." Therefore, the discriminant function $g(\mathcal{F}(Y;\Theta);\Lambda)$ of a given observation Y with a word string label Z_c is defined as

$$g\left(\mathcal{F}(Y;\Theta);\Lambda\right) = LL_c\left(\mathcal{F}(Y;\Theta);\Lambda\right) \tag{16}$$

where $LL_c(\mathcal{F}(Y;\Theta);\Lambda)$ represents the log-likelihood of the current enhanced feature vector sequence $\hat{X} = \mathcal{F}(Y;\Theta)$ under the current HMM parameters Λ against word string Z_c. The antidiscriminant function $\bar{g}(\cdot)$ is defined by using the N-best competitive word strings $\{Z_n\}_{n=1}^{N}$ other than the Z_c for each training utterance as follows:

$$\bar{g}\left(\mathcal{F}(Y;\Theta);\Lambda\right) = \frac{1}{\eta} log\left\{\frac{1}{N}\sum_{n=1}^{N} exp\left[\eta \cdot LL_n\left(\mathcal{F}(Y;\Theta);\Lambda\right)\right]\right\}$$

where η is a positive control parameter, and $LL_n(\cdot)$ represents the log-likelihood of $\mathcal{F}(Y;\Theta)$ against the word string Z_n.

As mentioned previously, both the mapping function parameters Θ and the HMM parameters Λ will be jointly updated using the following sequential gradient descent algorithm. Let us use Γ to denote generically the parameters to be estimated $\{\Theta, \Lambda\}$. Given $\mathcal{Y}$, we first randomize the ordering of $\{Y_i\}$, and then we present the training samples sequentially. Upon the presentation of the jth training sample, Γ is updated as follows:

$$\Gamma_{j+1} = \Gamma_j - \epsilon_j V_j \nabla l\left(\mathcal{F}(Y_j;\Theta);\Lambda\right)\big|_{\Gamma=\Gamma_j} \tag{17}$$

where "j" represents the cumulative number of training samples presented so far, V_j is a positive definite scaling matrix, and ϵ_j is the learning rate. One pass of the training samples is called an epoch. After the completion of each epoch, one could randomize the ordering of $\{Y_i\}$ again, to decrease the chance of being trapped in a local optimum with the same order of learning sequences. However, for the sake of simplicity, this step was skipped in our experiments. The learning rate ϵ_j is an important control parameter in MCE training. In this study, the following schedule is used:

$$\epsilon_{j+1} = \epsilon_j - \frac{\epsilon_0 \cdot T_j}{\sum_{j=1}^{EP\cdot I} T_j} \tag{18}$$

where T_j is the frame number of the jth cumulative training sentence, EP is the total number of training epoches to be performed in MCE training, and ϵ_0 is a control parameter need to be carefully determined by experiments. The scaling matrix V_j is set according to the method described in [20]. In all of the MCE training experiments to be reported in the next section, the following setting is used for the relevant control parameters: $\alpha = 0.05$, $\beta = 0$, $\eta = 0.05$, $N = 8$, $\epsilon_0 = 2$.

In order to find the gradient ∇l, the following partial derivative is used:

$$\frac{\partial l}{\partial \Gamma} = \alpha l(1-l)\left[-\frac{\partial LL_c}{\partial \Gamma} + \sum_{n=1}^{N}\frac{\exp(\eta \cdot LL_n)\frac{\partial LL_n}{\partial \Gamma}}{\sum_{n'=1}^{N}\exp(\eta \cdot LL_{n'})}\right]. \tag{19}$$

The remaining partial derivative $\partial LL_c/\partial\Gamma$ (or $\partial LL_n/\partial\Gamma$) is formulated differently depending on the parameters to be optimized. Since the process of updating HMM parameters Λ has been described in [20], only the formula related to the updating of Θ is presented in the following (LL denotes LL_c or LL_n generically)

$$\frac{\partial LL}{\partial \Theta} = -\sum_t\sum_s\sum_m \zeta_{jt}(s,m) \frac{\partial \mathcal{F}(y_{jt};\Theta)}{\partial \Theta}\Sigma_{sm}^{-1}\left(\mathcal{F}(y_{jt};\Theta) - \mu_{sm}\right). \tag{20}$$

In the above equation, $\zeta_{jt}(s,m)$ is the occupation probability of Gaussian component m in state s, at time t of current enhanced observation $\mathcal{F}(y_{jt};\Theta)$. For each $b_k^{(e)}$, from (2), it follows that

$$\frac{\partial \mathcal{F}(y_{jt};\Theta)}{\partial b_k^{(e)}} = \begin{cases} p(k|y_{jt},e), & \text{if } y_{jt} \text{ belongs to environment class } e \\ 0, & \text{otherwise} \end{cases}.$$

Therefore

$$\frac{\partial LL}{\partial b_k^{(e)}} = -\sum_t\sum_s\sum_m \zeta_{jt}(s,m)p(k|y_{jt},e) \Sigma_{sm}^{-1}\left[y_{jt} + \sum_{k'} p(k'|y_{jt},e)b_{k'}^{(e)} - \mu_{sm}\right] \tag{21}$$

if Y_j belongs to environment class e.

IV. Experiments and Results

A. Aurora2 Database

In order to verify the effectiveness of the proposed approach, a series of experiments are performed for the task of speaker independent recognition of connected digit strings on Aurora2 database. A full description of the Aurora2 database and a test framework is given in [16], and a brief description is given in the following to help readers understand better our experiments.

All speech utterances in Aurora2 are derived from the *TIDigits* database [22] with an original high-quality recording by downsampling to 8 kHz, filtering with G.712 or modified intermediate response system (MIRS) filter, and the controlled addition of noise to cover a wide range of signal-to-noise ratios (SNRs) under eight interesting noise conditions, namely, Suburban Train (denoted shortly as *Sub* hereinafter), Crowd of People (*Bab*), Car, Exhibition Hall (*Exh*), Restaurant (*Res*), Street (*Str*), Airport (*Apt*), and Train Station (*Sta*). According to [16], two training modes are defined as follows:

- *clean-condition*: training on clean data only;
- *multicondition*: training on clean and noisy data.

For the first mode, 8440 sentences are selected from the training part of *TIDigits* containing the recordings of 55 males and 55 females. They are filtered with the G.712 filter without adding noise. For the second mode, the same 8440 utterances are divided into 20 equally sized subsets, each containing a few utterances from all training speakers. The 20 subsets are distorted with four kinds of noises, including *Sub*, *Bab*, *Car*, and *Exh*, at five different SNRs, namely, clean condition, 20, 15, 10, and 5 dB. Again, the speech and noise are filtered with the G.712 filter before adding.

According to [16], three different test sets are also defined to simulate three different application scenarios. 4004 utterances from 52 male and 52 female speakers in the *TIDigits* test part are split into four subsets with 1001 utterances in each. Recordings of all speakers are present in each subset. *Test Set A* is formed by adding the same four types of noise signals as used in generating the multicondition training data to four subsets of testing utterances at SNRs of 20, 15, 10, 5, 0, and -5 dB, respectively. By further including the set of clean testing utterances before adding noises, the *Test Set A* consists of $4 \times 7 \times 1001 = 28\,028$ utterances that match the conditions of multicondition training data. The creation of *Test Set B* is exactly same as *Test Set A* except for using four different kinds of noises, i.e., *Res*, *Str*, *Apt*, and *Sta*. Both speech and noise signals in test sets A and B are filtered by G.712 filter before adding. Therefore, the *Test Set B* can be used to simulate application scenarios in the presence of distortions caused by mismatched additive noises only. *Test Set C* consists of two of the four subsets, which are distorted by noise signals of *Sub* and *Str*, respectively. This time, speech and noise are filtered by MIRS filter before adding them at SNRs of 20, 15, 10, 5, 0, and -5 dB, respectively. Again, by further including the clean testing utterances in the above two subsets, *Test Set C* consists of $2 \times 7 \times 1001 = 14\,014$ utterances that can be used for performance evaluation under a condition of mismatched channel only, or a condition with both mismatched channel and additive noises.

B. Baseline Systems

The recognition results presented in this section are produced on the Aurora2 database using two types of front-ends: the modified reference of Aurora front-end WI007 [12], [16] and the advanced front-end proposed in [13]. In the original WI007 front-end, for each frame, a 39-dimensional feature vector is generated, which consists of 12 mel frequency cepstral coefficients (MFCCs) (MFCC of order 0 is not included) and logarithmic frame energy, plus their first and second-order derivatives. The major modification on WI007 front-end in this evaluation is to use MFCC of order 0 but not log-energy for each frame. Another modification on WI007 is that the cepstra are computed based on the power spectral density instead of the magnitude spectrum. It was shown by experiments that the modified front-end is better than WI007 at the presence of noise distortion. Hereinafter, this modified version of WI007 front-end is called the *standard front-end*.

The *advanced front-end* is proposed in [13] as standard of the activity for distributed speech recognition by ETSI. This standard describes the front-end algorithm based on mel-cepstral feature extraction technique. It includes the feature extraction module and the feature compression/transmission part at both the terminal and server sides. In the feature extraction part, a two-stage mel-warped Wiener filter is designed and performed for noise reduction first. Then, the waveform processing is applied to the denoised signal and cepstral features are calculated. Last, the blind equalization is applied to the cepstral features, which are then fed to the further compression process for channel transmission. In our evaluation, we ignore all the operations of that standard at the server side in the distributed speech recognition. In other words, we will only use the cepstral features and their derivatives calculated in the feature extraction module as the front-end for Aurora2 evaluation.

The experiments are designed to recognize ten digits from "zero" to "nine," plus an "oh" which is another possible pronunciation of the digit "0." Whole word left-to-right CDHMMs are created for all digits. The CDHMM of each word consists of 16 emitting states, each having 20 Gaussian mixture components with diagonal covariance matrices, and two dummy states at the beginning and end for the easy implementation with HTK [40]. Besides, two pause models, "sil" and "sp," are created to model the silence before/after the digit string and the short pause between any two digits. The model of "sil" is defined as a five-state (including two dummy states) CDHMM with a flexible transition structure [16]. Each emitting state is modeled by a mixture of 36 Gaussian components. The model of "sp" consists of two dummy states and a single emitting state which is tied with the middle state of "sil."

During recognition, an utterance can be modeled by any sequence of digits with the possibility of a "sil" model at the beginning and at the end and an "sp" model between any two digits. All of the recognition experiments are performed with the search engine of HTK3.0 toolkit [40].

The experimental results reported in this paper only include those achieved under multicondition training of Aurora2 database and tested on all of three test sets. The performance (word accuracy in %) of our baseline systems using standard front-end

and advanced front-end are listed in Table V and VI of the Appendix, respectively. To compare and evaluate different approaches described in this paper, for each type of noise, a performance measure is introduced as an average over SNRs between 0 and 20 dB. The overall performance measure is averaged over all noises and over SNRs between 0 and 20 dB. For example, as shown in Table V, the average accuracy over Test Set A using standard front-end is 91.93%, while the overall average accuracy over three test sets is 90.96%. It is observed from the results that the advanced front-end removes much distortion in both training and testing noisy utterances, thus it is much more robust than the system using standard front-end, especially for the mismatched conditions, i.e., test set B. On average, the word error rate is reduced by 28.54% with advanced front-end in comparison with that of standard front-end.

C. Performance of SPLICE-Based Approaches Using Stereo Data

In this set of experiments, the SPLICE algorithm using available stereo data is implemented to construct a reference system for comparison with our joint design approach. In this reference system, 17 GMMs, each having a mixture of 256 Gaussian components with diagonal covariance matrices as described in (1), are trained for clean speech and the noisy speech under each of four noise types (*Sub*, *Bab*, *Car*, and *Exh*) and four SNR levels (20, 15, 10, and 5 dB). Sentence-based cepstral mean normalization (CMN) is performed before the GMM training as well as before the feature compensation.

The correction vectors for the full feature vectors, including the static features as well as the dynamic features, are estimated according to (4) and then used to calculate the pseudoclean feature vectors. The procedure of noise mean normalization (NMN) described in [11] is not implemented in our reference system, although it is reported by Microsoft researchers that the performance in mismatched conditions can be improved by doing so. Indeed, since the so-called NMN procedure cannot iteratively estimate the distortion on the dynamic cepstral features caused by the additive noise, it can just be integrated with the stochastic vector mapping approach where the transformation is merely applied on static features. However, by experiments, it is observed that such a system cannot even outperform the reference system of applying SPLICE on the full feature vector, as implemented in this study. Therefore, the NMN procedure is skipped in our experiments.

As mentioned in Section II-B, the CDHMMs used to recognize the digit strings can be trained by an ML principle or MCE principle over all of the pseudoclean feature vectors enhanced by SPLICE algorithm. The results of the reference systems exploiting both training schemes and using the standard front-end are summarized as average word error rates (in %) on three test sets in Table I, in which SPLICE stands for the experiments of collectively training the CDHMMs by ML criterion and SPLICE-MCEH stands for its MCE training counterpart (for HMM means only). Five epoches are conducted for MCE training. The experimental results show that the overall error rate is reduced from 9.04% of that without any noise compensation to 7.21% by using SPLICE and ML trained CDHMMs. The relative error rate reduction is about 20.2%. From Table I, we can also find that the SPLICE works much better in the condition where stereo training data exists (i.e., Test Set A) than in unseen condition (i.e., Test Set B). Apparently, the correction vectors estimated by ML criterion with stereo data are not representative enough to enhance the noisy speech in unseen environment and the ML-trained HMMs of different classes seem not discriminative enough to warrant a good performance. By simply training CDHMMs by MCE principle, i.e., SPLICE-MCEH, which is a natural extension of above experiments, the word error rate can be further reduced to 6.88%.

Similar experiments are also carried out on the advanced front-end, of which the results are summarized in Table II. Although the word accuracies of the baseline system using advanced front-end have been relatively high, the approach of "SPLICE-MCEH" can still reduce the word error rate by 12.23%, which demonstrates the effectiveness of MCE training.

TABLE I
SUMMARY OF DIGIT RECOGNITION ERROR RATES OF STOCHASTIC VECTOR MAPPING APPROACHES USING STEREO DATA (STANDARD FRONT-END)

	Set A	Set B	Set C	Overall
Baseline	8.07%	9.40%	10.30%	9.04%
SPLICE	6.22%	8.35%	6.90%	7.21%
SPLICE-MCEH	5.90%	8.03%	6.57%	6.88%

TABLE II
SUMMARY OF DIGIT RECOGNITION ERROR RATES OF STOCHASTIC VECTOR MAPPING APPROACHES USING STEREO DATA (ADVANCED FRONT-END)

	Set A	Set B	Set C	Overall
Baseline	5.91%	6.68%	7.11%	6.46%
SPLICE	5.12%	6.71%	6.02%	5.95%
SPLICE-MCEH	4.90%	6.34%	5.88%	5.67%

TABLE III
SUMMARY OF DIGIT RECOGNITION ERROR RATES OF STOCHASTIC VECTOR MAPPING APPROACHES WITHOUT USING STEREO DATA (ADVANCED FRONT-END)

	Set A	Set B	Set C	Overall
Baseline	5.91%	6.68%	7.11%	6.46%
Joint-ML	5.39%	6.41%	6.21%	5.96%
MCEH	5.65%	6.04%	6.12%	5.93%
Joint-MCE	5.17%	5.98%	5.99%	5.66%

TABLE IV
PERFORMANCE COMPARISON IN TERMS OF AVERAGE DIGIT RECOGNITION ERROR RATES ON TEST SET A FOR TWO TYPES OF SYSTEMS WITH A SIMILAR NUMBER OF MODEL PARAMETERS: CDHMM VERSUS SVM-CDHMM (ADVANCED FRONT-END)

Systems with a Similar Number of Model Parameters	Training Approaches	
	ML	MCE
CDHMM (32 mixtures per state)	5.78%	5.56%
SVM-CDHMM (20 mixtures per state)	5.39%	5.17%

TABLE V
PERFORMANCE (WORD ACCURACY IN %) OF BASELINE SYSTEM (STANDARD FRONT-END)

	Set A					Set B				
	Sub	Bab	Car	Exh	Ave.	Res	Str	Apt	Sta	Ave.
Clean	99.48	99.40	99.49	99.54	99.48	99.48	99.40	99.49	99.54	99.48
20 dB	98.99	99.06	99.28	98.89	99.06	98.93	98.91	98.54	98.73	98.78
15 dB	98.50	98.61	99.02	98.49	98.66	98.28	98.13	97.73	97.56	97.92
10 dB	97.51	97.31	97.55	96.64	97.26	96.13	95.59	95.29	94.94	95.49
5 dB	93.71	91.75	93.02	91.14	92.41	90.27	89.48	90.43	88.46	89.67
0 dB	76.88	68.59	69.07	74.79	72.29	71.51	70.37	73.49	69.24	71.17
-5 dB	36.05	31.41	26.30	34.43	32.00	36.26	35.43	38.50	31.22	35.38
Ave.	93.12	91.06	91.59	91.99	91.93	91.02	90.50	91.10	89.79	90.60

	Set C			
	Sub M	Str M	Ave.	Overall
Clean	99.36	99.37	99.37	99.45
20 dB	99.02	98.76	98.89	98.91
15 dB	98.53	97.88	98.20	98.27
10 dB	96.93	96.10	96.51	96.40
5 dB	91.10	89.18	90.13	90.86
0 dB	62.36	67.11	64.75	70.34
-5 dB	24.99	30.50	27.77	32.51
Ave.	89.59	89.81	89.70	90.96

D. Performance of Joint Design Approaches Without Using Stereo Data

In order to examine the efficacy of the joint design approaches for training correction vectors and CDHMMs, two sets of experiments are performed by ML training (labeled as "Joint-ML") and MCE training (labeled as "Joint-MCE") of all the parameters. In all of the experiments, the same number of correction vectors and the same environment dependent GMM for environment identification as that of the experiments with stereo data are used. The "Joint-ML" approach is to estimate the correction vectors and CDHMMs from nonstereo clean/noisy training utterances. The values of mapping function parameters and HMM parameters in the "Joint-ML" system are also used to initialize the "Joint-MCE" training procedure. In the experiments of "Joint-MCE," eight best competitive word strings are generated by using the "Joint-ML" trained pseudoclean HMMs for each pseudoclean training utterance enhanced by stochastic vector mapping. For simplicity, the optimization of mapping function parameters and the HMM parameters are not performed simultaneously but alternately. At the beginning, the ML-trained parameters are treated as the initial values of the first iteration. The MCE training of correction vectors with five epoches for stochastic vector mapping is performed firstly. Then, starting from the newly estimated correction vectors and the initial ML-trained pseudoclean HMMs, the new HMM parameters (mean vectors only) are estimated by MCE training with five epoches. Finally, the resulted correction vectors and the pseudo-clean HMMs will be used as the initial values of the next iteration. The training process can thus continue as described above. In our experiments, we used only one iteration. From the above description, it can be seen that all of the experiments in this subsection do not need any stereo data.

Experiments of using the above two schemes are conducted for both standard front-end and advanced front-end. The trends of word error rate reduction are similar for both front-ends, thus only experimental results for advanced front-end are summarized in Table III. It is observed in Table III that "Joint-MCE" can achieve an error rate reduction of 12.4% in comparison with the baseline system performance. This performance improvement is similar to what was achieved by "SPLICE-MCEH" using stereo data as shown in Table II. By examining the detailed results listed in Tables II and III, it is not surprising that the system using stereo data can provide better performance under matched conditions, i.e., Test Set A, than that without using stereo data. However, the systems trained by using the joint design approaches (both ML and MCE) without using stereo data shows their ability in tolerating somehow mismatched testing conditions, i.e., Test Set B, such that the overall performances of two types of systems using and not using stereo data are almost the same. The effectiveness of the joint design approaches proposed in this paper is confirmed.

In order to further demonstrate the effect of feature compensation module based on stochastic vector mapping, the performance of MCE trained CDHMMs without using stochastic vector mapping-based feature compensation is also studied, which is shown as "MCEH" in Table III. Although the MCE trained CDHMMs outperform the ML trained CDHMMs in baseline system, they are worse than the "Joint-MCE" trained CDHMMs when the stochastic vector mapping is used to compensate for the noisy speech. This is consistent with our findings in [36], where more detailed results were reported in a different experimental setup on the effects of using MCE training for stochastic vector mapping parameters only, or HMM parameters only, or both. Readers are also referred to [38] for more experimental results on Aurora3 task.

TABLE VI
PERFORMANCE (WORD ACCURACY IN %) OF BASELINE SYSTEM (ADVANCED FRONT-END)

	Set A					Set B				
	Sub	Bab	Car	Exh	Ave.	Res	Str	Apt	Sta	Ave.
Clean	99.51	99.40	99.58	99.48	99.49	99.51	99.40	99.58	99.48	99.49
20 dB	99.32	99.21	99.46	99.29	99.32	99.42	99.15	99.40	99.66	99.41
15 dB	98.83	98.85	99.08	98.61	98.84	98.83	98.58	98.81	99.17	98.85
10 dB	97.45	97.55	98.12	97.19	97.58	96.71	96.95	97.32	97.38	97.09
5 dB	94.29	93.29	95.20	93.18	94.00	91.80	92.11	94.07	92.84	92.71
0 dB	82.28	74.18	85.24	81.09	80.70	74.30	78.87	80.61	80.38	78.55
-5 dB	51.80	37.94	52.34	53.38	48.84	39.30	48.40	48.14	51.19	46.77
Ave.	94.43	92.85	95.42	93.87	94.09	92.21	93.13	94.04	93.89	93.32

	Set C			
	Sub M	Str M	Ave.	Overall
Clean	99.48	99.46	99.47	99.49
20 dB	99.29	99.09	99.19	99.33
15 dB	98.83	98.76	98.79	98.84
10 dB	97.42	96.89	97.15	97.30
5 dB	93.55	91.75	92.64	93.21
0 dB	78.32	75.03	76.66	79.04
-5 dB	44.52	41.29	42.89	46.82
Ave.	93.48	92.30	92.89	93.54

E. Discussion

In the above systems based on stochastic vector mapping, model parameters consist of two parts: SVM parameters Θ and CDHMM parameters Λ. The total number of parameters in SVM-CDHMM systems is more than that in a CDHMM-based baseline system. It would be interesting to see whether an SVM-CDHMM system can still perform better than a CDHMM-based system with a similar number of model parameters. Another set of experiments are conducted to verify this issue. We simply increase the number of Gaussian mixture components per state from 20 to 32 for each digit CDHMM, and from 36 to 48 for "sil" CDHMM. Such a CDHMM-based system has a similar number of model parameters with that of an SVM-CDHMM system in the previous experiments. Two new CDHMM-based baseline systems are trained by using ML and MCE training, respectively. Their performances in terms of average digit recognition error rates on Test Set A are compared with that of the corresponding SVM-CDHMM systems with a similar number of model parameters in Table IV. It is observed that the SVM-CDHMM systems outperform the traditional CDHMM systems. This confirms again the usefulness of using the SVM for environment compensation.

Another concern of an SVM-CDHMM system is the increased computational complexity during recognition stage. To get a clear idea, a timing experiment is conducted on a "Pentium-4" PC with a 2.4-GHz clock by using a randomly selected testing sentence with a length of 2.8 s. The *user CPU time* for a full Viterbi decoding of the above sentence in our baseline system (i.e., 20 mixtures per state for each digit CDHMM) is 0.63 s. The *user CPU time* for recognizing the above sentence in our SVM-CDHMM system is 1.33 s that include 0.64 s for "environment condition" labeling, 0.06 s for feature vector mapping, and 0.63 s for Viterbi decoding of the "denoised" sentence. The main overhead comes from the "environment condition" labeling because a large number of Gaussian components are involved. How to speed up the relevant operations is a topic for further research.

V. SUMMARY

In this paper, we have presented an environment-compensated MCE training approach for the joint design of the feature compensation module and HMM parameters of a speech recognizer. The feature compensation module is based on a stochastic vector mapping function whose parameters have to be learned from stereo data in a previous approach called SPLICE. In our proposed MCE joint design approach, by initializing the parameters with an approximate ML training procedure, the requirement of stereo data can be removed without causing performance degradation as demonstrated in benchmark evaluations on Aurora2 connected digits database. Although the baseline system using the ETSI advanced front-end has achieved an overall word error rate of 6.46% on Aurora2 benchmark test sets, an additional 12.4% error rate reduction is achieved by using our proposed MCE training approach based on stochastic vector mapping. As a future work, we will study its effectiveness in a subword-based large vocabulary ASR application.

APPENDIX I
FULL RESULTS OF BASELINE SYSTEMS WITH STANDARD AND ADVANCED FRONT-ENDS

See Tables V and VI.

REFERENCES

[1] A. Acero, *Acoustic and Environment Robustness in Automatic Speech Recognition*. Norwell, MA: Kluwer, 1993.
[2] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, "A compact model for speaker-adaptive training," in *Proc. ICSLP*, 1996, pp. 1137–1140.

[3] A. Biem and S. Katagiri, "Feature extraction based on minimum classification error/generalized probabilistic descent method," in *Proc. ICASSP*, 1993, pp. II-275–II-278.
[4] A. Biem, S. Katagiri, E. McDermott, and B.-H. Juang, "An application of discriminative feature extraction to filter-bank-based speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 9, no. 2, pp. 96–110, Mar. 2001.
[5] R. Chengalvarayan and L. Deng, "HMM-based speech recognition using state-dependent, discriminatively derived transforms on Mel-Warped DFT features," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 243–256, May 1997.
[6] R. Chengalvarayan, "Evaluation of front-end features and noise compensation methods for robust Mandarin speech recognition," in *Proc. Eurospeech*, Aalborg, Denmark, 2001, pp. 897–900.
[7] W. Chou, M. G. Rahim, and E. Buhrke, "Signal conditioned minimum error rate training," in *Proc. Eurospeech*, 1995, pp. 495–498.
[8] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," in *J. Roy. Statist. Soc. Ser. B*, 1977, vol. 39, no. 1, pp. 1–38.
[9] L. Deng, A. Acero, M. Plumpe, and X.-D. Huang, "Large-vocabulary speech recognition under adverse acoustic environments," in *Proc. ICSLP*, Oct. 2000, pp. III-806–809.
[10] L. Deng, A. Acero, L. Jiang, J. Droppo, and X.-D. Huang, "High-performance robust speech recognition using stereo training data," in *Proc. ICASSP*, 2001, pp. I-301–I-304.
[11] L. Deng, J. Droppo, and A. Acero, "Recursive estimation of nonstationary noise using iterative stochastic approximation for robust speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 568–580, Nov. 2003.
[12] J. Droppo, L. Deng, and A. Acero, "Evaluation of the SPLICE algorithm on the Aurora2 database," in *Proc. Eurospeech*, Aalborg, Denmark, Sep. 2001, pp. 217–220.
[13] *Speech Processing, Transmission and Quality Aspects (STQ); Distributed Speech Recognition; Advanced Front-End Feature Extraction Algorithm; Compression Algorithms*, ETSI ES 202 050 v1.1.1 (2002-10), Oct. 2002, ETSI standard document.
[14] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," in *Comput. Speech Lang.*, 1998, vol. 12, pp. 75–98.
[15] Y. Gong, "Source normalization training for HMM applied to noisy telephone speech recognition," in *Proc. Eurospeech*, 1997, pp. 1555–1558.
[16] H. G. Hirsch and D. Pearce, "The AURORA experimental framework for the performance evaluations of speech recognition systems under noisy conditions," in *ISCA ITRW ASR*, Paris, France, Sep. 2000, pp. 181–188.
[17] W.-T. Hong and S.-H. Chen, "A robust training algorithm for adverse speech recognition," *Speech Commun.*, vol. 30, no. 4, pp. 273–293, 2000.
[18] W.-T. Hong, "A discriminative and robust training algorithm for noisy speech recognition," in *Proc. ICASSP*, 2003, pp. I-8–I-11.
[19] B.-H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. Signal Process.*, vol. 40, no. 12, pp. 3043–3054, Dec. 1992.
[20] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 257–265, May 1997.
[21] C. Lawrence and M. Rahim, "Integrated bias removal techniques for robust speech recognition," in *Comput. Speech . Lang.*, 1999, vol. 13, pp. 283–298.
[22] R. G. Leonard, "A database for speaker-independent digit recognition," in *Proc. ICASSP*, 1984, pp. 42.11.1–42.11.4.
[23] R. P. Lippmann, E. A. Martin, and D. B. Paul, "Multi-style training for robust isolated-word speech recognition," in *Proc. ICASSP*, 1987, pp. 705–708.
[24] B. Mak, Y.-C. Tam, and P. Li, "Discriminative auditory-based features for robust speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 12, no. 1, pp. 27–36, Jan. 2004.
[25] J. McDonough, T. Schaaf, and A. Waibel, "On maximum mutual information speaker-adapted training," in *Proc. ICASSP*, 2002, pp. I-601–I-604.
[26] P. Moreno, "Speech Recognition in Noisy Environments," Ph.D. dissertation, Dept. Elect. Comput. Eng., Carnegie Mellon Univ., Pittsburgh, PA, 1996.
[27] K. K. Paliwal, M. Bacchini, and Y. Sagisaka, "Simultaneous design of feature extractor and pattern classifier using the minimum classification error training algorithm," in *Proc. NNSP*, 1995, pp. 67–76.
[28] M. Rahim and C.-H. Lee, "Simultaneous feature and HMM design using string-based minimum classification error training criterion," in *Proc. ICSLP*, 1996, pp. 1820–1823.
[29] M. Rahim, B.-H. Juang, W. Chou, and E. Buhrke, "Signal conditioning techniques for robust speech recognition," *IEEE Signal Process. Lett.*, vol. 3, no. 4, pp. 107–109, Apr. 1996.
[30] A. Sankar and C.-H. Lee, "A maximum-likelihood approach to stochastic matching for robust speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 4, no. 3, pp. 190–202, May 1996.
[31] R. Singh, R. M. Stern, and B. Raj, "Signal and feature compensation methods for robust speech recognition," in *Noise Reduction Speech Applications*, G. M. Davis, Ed. Boca Raton, FL: CRC, 2002, pp. 219–244.
[32] R. M. Stern, A. Acero, F.-H. Liu, and Y. Ohshima, "Signal processing for robust speech recognition," in *Automatic Speech and Speaker Recognition: Advanced Topics*, C.-H. Lee, F. Soong, and K. K. Paliwal, Eds. Norwell, MA: Kluwer, 1996, pp. 351–378.
[33] A. Torre, A. M. Peinado, A. J. Rubio, V. E. Sanchez, and J. E. Diaz, "An application of minimum classification error to feature space transformations for speech recognition," *Speech Commun.*, vol. 20, pp. 273–290, 1996.
[34] S. Tsakalidis, V. Doumpiotis, and W. Byrne, "Discriminative linear transforms for feature normalization and speaker adaptation in HMM estimation," in *Proc. ICSLP*, 2002, pp. 2585–2588.
[35] L. Wang and P. C. Woodland, "Discriminative adaptive training using the MPE criterion," in *Proc. ASRU*, 2003, pp. 279–284.
[36] J. Wu and Q. Huo, "An environment compensated minimum classification error training approach and its evaluation on Aurora2 database," in *Proc. ICSLP*, Denver, CO, 2002, pp. I-453–I-456.
[37] ——, "Several HKU approaches for robust speech recognition and their evaluation on Aurora connected digit recognition tasks," in *Proc. Eurospeech*, Geneva, Switzerland, 2003, pp. 21–24.
[38] J. Wu, Q. Huo, and D.-L. Zhu, "An environment compensated maximum likelihood training approach based on stochastic vector mapping," in *Proc. ICASSP*, Philadelphia, PA, 2005, pp. I-429–I-432.
[39] J.-X. Wu, Z. Qi, C. Chan, and J. Li, "Speaker normalization by input space optimization for continuous density hidden Markov models," in *1994 Int. Symp. Speech, Image Process. Neural Netw.*, Hong Kong, China, Apr. 1994, pp. 682–685.
[40] S. Young *et al.*, The HTK Book (for HTK V3.0) July 2000.

Jian Wu (M'05) received the B.S. and M.S. degrees in computer science from Tsinghua University, Beijing, China, in 1998 and 2000, respectively, and the Ph.D. degree in computer science from the University of Hong Kong (HKU), Hong Kong, China, in 2004.

From 1997 to 2000, he was with the Center of Speech Technology, Tsinghua University. From 2000 to 2004, he was a member of the Human Machine Communication Laboratory, HKU. In 2004, he joined the Speech and Natural Language Group, Microsoft Corporation, Redmond, WA, working on speech recognition for desktop, telephony server, and other devices. His current research interests include speech recognition in adverse acoustical environment, acoustic modeling, front-end processing, and machine learning for speech recognition.

Qiang Huo (M'95) received the B.Eng. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1987, the M.Eng. degree from Zhejiang University, Hangzhou, China, in 1989, and the Ph.D. degree from the USTC in 1994, all in electrical engineering.

From 1986 to 1990, his research work focussed on the hardware design and development for real-time digital signal processing, image processing and computer vision, and speech and speaker recognition. From 1991 to 1994, he was with the Department of Computer Science, the University of Hong Kong (HKU), Hong Kong, China, where he conducted research on speech recognition. From 1995 to 1997, he was with ATR Interpreting Telecommunications Research Laboratories, Kyoto, Japan, where he engaged in research in speech recognition. He rejoined the Department of Computer Science, HKU, in 1998 and has been an Associate Professor there since 2001. His current major research interests include automatic speech recognition; handwriting recognition; user authentication via voice, fingerprint, and their combination; Chinese/English OCR; computational model for spoken-language processing; multimodal human–machine communication; adaptive signal modeling and processing; general pattern recognition theory; machine learning; and information retrieval.