7th International Conference on Spoken Language Processing [ICSLP2002]
Denver, Colorado, USA
September 16-20, 2002

AN ENVIRONMENT COMPENSATED MINIMUM CLASSIFICATION ERROR TRAINING APPROACH AND ITS EVALUATION ON AURORA2 DATABASE

Jian Wu and Qiang Huo

Department of Computer Science and Information Systems
The University of Hong Kong, Pokfulam Road, Hong Kong, China
(Email: jwu@csis.hku.hk, qhuo@csis.hku.hk)

ABSTRACT

A conventional feature compensation module for robust automatic speech recognition is usually designed separately from the training of HMM parameters of the recognizer, albeit a maximum likelihood criterion might be used in both designs. In this paper, we present an environment compensated minimum classification error training approach for the joint design of the feature compensation module and the recognizer itself. By evaluating the proposed approach on Aurora2 connected digits database, a digit recognition error rate, averaged on all three test sets, of 9.15% and 13.98% is achieved for multi- and clean-condition training respectively. In comparison with the performance achieved by the baseline system without any environment compensation provided by the organizer of the ICSLP-2002 special session on Aurora tasks, our approach achieves an overall error rate reduction of 54.60%.

1. INTRODUCTION

It is well known that the performance of an automatic speech recognition (ASR) system will deteriorate 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 achieved under matched training and test conditions. Recently, Microsoft researchers [4, 5] developed a feature-based compensation algorithm, namely SPLICE (Stereo-based Piecewise Linear Compensation for Environment), and demonstrated its potential of surpassing such a limit.

For SPLICE and many other frame-dependent bias removal algorithms developed at CMU (e.g.,[12]) and some other places, in both training and testing, the noisy speech features are mapped into the clean speech features by a simple transformation, which is referred to as a *stochastic vector mapping* in this paper. After each training utterance is enhanced, a multi-style training is performed on all of the pseudo-clean feature vectors to estimate the HMM parameters of the recognizer. Apparently, the success of such a framework relies on 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.

This research was supported by grants from the RGC of the Hong Kong SAR (Project No. HKU7022/00E) and the ATR-SLT Lab of Japan.

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 can not guarantee to achieve the objective of minimum classification error (MCE) in recognition. It is thus well-motivated to use an MCE criterion for the *joint* training of the above two sets of parameters.

In [3], a signal conditioned MCE training approach was proposed and demonstrated to be effective for compensating the distortions caused by both the channel mismatch as well as additive noises [3, 2]. Although the HMM parameters are MCE-trained on the compensated feature vectors, the feature compensation module is derived directly from the HMM parameters under another criterion. Furthermore, it has been demonstrated by several researchers that MCE criterion can be beneficially applied to the design of a feature extractor [1] either separately from or jointly with the training of the recognizer parameters (e.g., [9, 10]).

Inspired by the previous works, in this paper, we propose an environment compensated MCE training approach for the *joint* design of the feature compensation module and the recognizer itself. The rest of the paper is organized as follows. In Section 2, we review the framework of the conventional ML-based stochastic vector mapping approach for environment compensation and establish the necessary notations. In Section 3, we describe the proposed approach. In Section 4, we report the evaluation results on Aurora2 database to demonstrate the effectiveness of the proposed approach. Finally, we conclude the paper in Section 5.

2. ML-BASED STOCHASTIC VECTOR MAPPING AND HMM DESIGN FOR ENVIRONMENT COMPENSATION

We assume that the speech signal corrupted by the additive noise and convolutional distortion has been transformed by signal processing operations into a vector of parameters (MFCCs and their derivatives in this paper). The task of feature-based environment 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 (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. 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 CMU's algorithms and SPLICE, is to approximate it by using a stochastic vector mapping approach as described in the following.

Given a set of training data $\mathcal{Y} = \{Y_i\}_{i=1}^{I}$, where Y_i is a sequence of feature vectors of noisy speech, suppose that they can be partitioned into E environments. Assume that the feature vector y under an environment e follows the distribution of a mixture of Gaussian PDFs (probability density function):

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

where $\mathcal{N}(\cdot; \xi, R)$ is a normal distribution with mean vector ξ and diagonal covariance matrix R. The above model parameters can be estimated easily from the corresponding set of training data by using the EM algorithm. Let's define the *stochastic vector mapping function* as follows:

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

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 the ML criterion [8]. If the stereo data is used to estimate $\Theta^{(e)}$, this becomes the SPLICE approach. In this way, each training environment e can be characterized by a Gaussian mixture model (GMM) as shown in Eq. (1) and a stochastic vector mapping function as shown in Eq. (2). If we transform each training feature vector accordingly, a pseudo-clean 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). Consequently, the set of pseudo-clean HMM parameters, Λ, can be estimated by using an ML-based multi-style training strategy.

In recognition, given an unknow utterance $Y = (y_1, y_2, \cdots, y_T)$, the most similar training condition is first identified as that of having the maximum likelihood $p(Y|e) = \prod_{t=1}^{T} p(y_t|e)$ for $e = 1, 2, \cdots, E$. Then the corresponding GMM and the mapping function are used to derive a pseudo-clean version of $\hat{X}$ from Y. For the convenience of notation, we also use hereinafter $\mathcal{F}(Y; \Theta)$ to denote the enhanced version of the utterance Y by transforming individual feature vector y_t as defined in Eq. (2). After feature compensation, $\hat{X}$ is finally recognized by a CDHMM recognizer trained as described above. Again, for simplicity, the superscript e in $\Theta^{(e)}$ is ignored in the following discussion.

3. MCE TRAINING OF STOCHASTIC VECTOR MAPPING FUNCTION PARAMETERS AND HMMS

In our proposed environment compensated MCE training approach, we still use the stochastic vector mapping function as defined in Eq. (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 set

$$\ell(\Theta, \Lambda) = \frac{1}{I} \sum_{i=1}^{I} l(\mathcal{F}(Y_i; \Theta); \Lambda) \ , \tag{4}$$

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

$$l(\mathcal{F}(Y; \Theta); \Lambda) = \frac{1}{1 + exp(-\alpha d(\mathcal{F}(Y; \Theta); \Lambda) + \beta)} \ , \tag{5}$$

where α and β are two control parameters. In the above equation, $d(\cdot)$ is a misclassification measure defined as

$$d(\mathcal{F}(Y; \Theta); \Lambda) = -g(\mathcal{F}(Y; \Theta); \Lambda) + \bar{g}(\mathcal{F}(Y; \Theta); \Lambda) \ , \tag{6}$$

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 (digit) 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

$$g(\mathcal{F}(Y; \Theta); \Lambda) = LL_c(\mathcal{F}(Y; \Theta); \Lambda) \ , \tag{7}$$

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:

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

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 above, both the mapping function parameters Θ and the HMM parameters Λ will be jointly updated using the following sequential gradient descent algorithm. Let's 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 j-th training sample, Γ is updated as follows:

$$\Gamma_{j+1} = \Gamma_j - \epsilon_j V_j \nabla l(\mathcal{F}(Y_j; \Theta); \Lambda)|_{\Gamma = \Gamma_j} \ , \tag{8}$$

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, we need to randomize the ordering of $\{Y_i\}$ again. In order to find the gradient ∇l, the following partial derivative is used,

$$\frac{\partial l}{\partial \Gamma} = \alpha l(1-l)[-\frac{\partial LL_c}{\partial \Gamma} + \sum_{n=1}^{N} \frac{\exp(\eta \cdot LL_n)\frac{\partial LL_n}{\partial \Gamma}}{\sum_{j=1}^{N} \exp(\eta \cdot LL_j)}] \ . \tag{9}$$

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 [7], only the formula related to the updating of Θ is presented in the following (LL denotes LL_c or LL_n generically):

$$\frac{\partial LL}{\partial \Theta} = -\sum_t \sum_s \sum_m \zeta_t(s,m) \Sigma_{sm}^{-1} (\mathcal{F}(y_t; \Theta) - \mu_{sm}) \frac{\partial \mathcal{F}(y_t; \Theta)}{\partial \Theta} \ . \tag{10}$$

In the above equation, $\zeta_t(s,m)$ is the occupation probability of Gaussian component m in state s, with mean vector μ_{sm} and diagonal covariance matrix Σ_{sm}, at time t of current observation. It can be calculated with a Forward-Backward procedure using current enhanced training vectors $\hat{X}$ against current HMM parameters Λ and the related word string. For each $b_k^{(e)}$, from Eq. (2), it follows that,

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

4. EXPERIMENTS AND RESULTS

4.1. Aurora2 Database and Experimental Setup

The task used in our study is the speaker independent recognition of connected digit strings. The recognition results presented in this section are produced on the Aurora2 database using the reference of Aurora front-end version 2.0 [6]. In this front-end, for each frame, a 39-dimensional feature vector is generated, which consists of 12 MFCCs (MFCC of order 0 is not included) and logarithmic frame energy, plus their first and second order derivatives. Whole word left-to-right CDHMMs are created for all digits. The CDHMM consists of 18 states, each having 3 Gaussian mixture components with diagonal covariance matrices. 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. All of the recognition experiments are performed with the search engine of HTK3.0 toolkit [13] and follow exactly the default scripts provided in Aurora2 CD-ROM.

The SPLICE algorithm is implemented to construct a reference system for comparison with our MCE-based approach. The values of mapping function parameters and/or HMM parameters in the reference system are also used to initialize the MCE training procedure. In this reference system, 17 GMMs, each having a mixture of 256 Gaussian components with diagonal covariance matrices, are trained for the noisy speech under each combination of noise type and level. Sentence based Cepstral Mean Normalization (CMN) is performed in both training and testing before feature vectors are enhanced. The procedure of Noise Mean Normalization (NMN) described in [5] is not implemented in our reference system, although it is observed by Microsoft researchers that the performance in mismatched conditions can be improved greatly by doing so. It would be an interesting future work to see what would happen if the NMN is combined with the MCE-based approach proposed in this paper.

4.2. Experimental Results

The results of the baseline system without any noise compensation [11] and the reference system using SPLICE are shown as “BASELINE" and “SPLICE" in Table 1, respectively. In comparison with the following experiments where HMM parameters Λ and/or mapping function parameters Θ are trained by MCE criterion, the experimental results reported in Table 1 only include those achieved under multi-condition training of Aurora2 database. It should be noted that the re-endpointed utterances [11] are used for both training and testing in “BASELINE" while not used in our own experiments.

The experimental results show that the overall error rate is reduced from 12.97% of that without any noise compensation to 11.83% by using ML-based feature vector mapping (SPLICE). The relative error rate reduction is about 9%. From Table 1, 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 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.

Table 1. Aurora2 Word Error Rate (Multicondition Training)

	Set A	Set B	Set C	Overall
BASELINE	11.93%	12.78%	15.44%	12.97%
SPLICE	9.99%	12.92%	13.34%	11.83%
MCE-VM	8.97%	12.22%	12.01%	10.88%
MCE-HMM	7.91%	11.92%	11.03%	10.03%
MCE-INTEG	7.08%	10.85%	9.91%	9.15%

In order to examine the efficacy of using MCE training, three sets of experiments are performed by MCE training the stochastic vector mapping function parameters only (labeled as “MCE-VM" in Table 1) or the HMM parameters only (labeled as “MCE-HMM" in Table 1) or both (labeled as “MCE-INTEG" in Table 1). In all of the experiments, eight best competitive word strings are generated by using the ML-trained pseudo-clean HMMs for each pseudo-clean training utterance enhanced by SPLICE. It can be observed in Table 1 that the performance of “MCE-HMM" in Test Set A is improved dramatically in comparison with SPLICE. Even for Test Set B, an obvious improvement is observed. The overall error rate becomes 10.03%. This reflects a relative error rate reduction of 23% in comparison with the baseline system. In comparison with the reference system with SPLICE, the relative error rate reduction is 15.2%. As for “MCE-VM", the overall error rate is reduced to 10.88%. Although it is not as significant as that of “MCE-HMM", it still demonstrates the usefulness of the MCE training of the mapping function parameters.

As for “MCE-INTEG", the mapping function parameters and the HMM parameters are both trained by using MCE criterion. For simplicity, the optimization of mapping function parameters and the HMM parameters are not performed simultaneously but alternately. In the beginning, the ML-trained parameters are treated as the initial values of the first iteration. The MCE training of correction vectors for stochastic vector mapping is performed firstly. Then, starting from the newly estimated correction vectors and the initial ML-trained pseudo-clean HMMs, the new HMM parameters are estimated with MCE training. 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. As a result, “MCE-INTEG" achieves the best performance among the above sets of experiments. The overall error rate is reduced by 29.5% and 22.7% in comparison with the baseline and reference systems, respectively.

Finally, as required by the organizer of the ICSLP-2002 special session on Aurora tasks [11], we perform the “MCE-INTEG" experiments for both the multi- and clean-condition training on Aurora2 database. By using such an integrated optimization, the best experimental results we achieved are shown in Table 2 and Table 3 using the prescribed formats.

Table 2. Absolute Results of Joint Design of Stochastic Vector Mapping and HMM(%)

Multicondition training, multicondition testing														
	A					B					C			
	Sub	Bab	Car	Exh	Ave.	Res	Str	Apt	Sta	Ave.	Sub M	Str M	Ave.	Overall
Clean	99.51	99.21	99.49	99.48	99.42	99.51	99.21	99.49	99.48	99.42	99.39	99.30	99.35	99.41
20 dB	99.23	98.88	99.40	99.26	99.19	98.93	98.67	99.25	99.23	99.02	99.29	98.55	98.92	99.07
15 dB	98.89	98.52	98.81	98.46	98.67	98.07	97.67	98.06	97.62	97.86	98.74	97.55	98.15	98.24
10 dB	97.64	96.40	97.94	97.41	97.35	95.67	95.98	95.85	94.60	95.53	97.05	94.44	95.75	96.30
5 dB	94.78	89.78	93.50	93.34	92.85	86.55	89.24	88.97	86.79	87.89	93.09	83.80	88.45	89.98
0 dB	82.99	63.81	78.85	80.47	76.53	62.48	67.41	65.73	66.21	65.46	79.18	59.25	69.22	70.64
-5 dB	49.55	25.82	32.30	45.51	38.30	24.26	29.93	28.18	26.94	27.33	43.94	27.06	35.50	33.35
Ave.	94.71	89.48	93.70	93.79	92.92	88.34	89.79	89.57	88.89	89.15	93.47	86.72	90.09	90.85

Clean training, multicondition testing														
	A					B					C			
	Sub	Bab	Car	Exh	Ave.	Res	Str	Apt	Sta	Ave.	Sub M	Str M	Ave.	Overall
Clean	99.39	99.27	99.49	99.44	99.40	99.39	99.27	99.49	99.44	99.40	99.60	99.24	99.42	99.40
20 dB	98.68	98.52	98.84	98.52	98.64	98.62	98.19	98.96	98.70	98.62	98.16	98.16	98.16	98.54
15 dB	97.64	97.37	97.70	97.10	97.45	97.76	96.89	97.73	97.10	97.37	96.71	95.95	96.33	97.20
10 dB	94.66	93.80	95.38	93.86	94.43	93.89	93.44	94.90	93.61	93.97	92.97	89.66	91.30	93.62
5 dB	87.72	82.56	86.22	85.47	85.49	80.66	82.65	83.30	81.98	82.16	84.25	73.04	78.60	82.78
0 dB	70.03	52.75	63.73	65.91	63.07	52.20	55.38	55.92	54.83	54.60	62.08	46.92	54.44	57.95
-5 dB	38.87	19.01	28.42	35.82	30.46	16.27	21.28	21.92	18.57	19.54	34.57	21.22	27.84	25.57
Ave.	89.75	85.00	88.37	88.17	87.82	84.63	85.31	86.16	85.24	85.34	86.83	80.75	83.77	86.02

Table 3. Aurora2 Summaries of Joint Design

Aurora 2 Word Error Rate				
	Set A	Set B	Set C	Overall
Multi	7.08%	10.85%	9.91%	9.15%
Clean	12.18%	14.66%	16.21%	13.98%
Average	9.63%	12.76%	13.06%	11.57%

Aurora 2 Relative Improvement				
	Set A	Set B	Set C	Overall
Multi	47.23%	29.18%	39.27%	38.42%
Clean	72.57%	74.35%	60.11%	70.79%
Average	59.90%	51.77%	49.69%	54.60%

5. SUMMARY

We have presented an environment compensated MCE training approach for the joint design of the feature compensation module and the recognizer itself. By evaluating the proposed approach on Aurora2 connected digits database, a digit recognition error rate, averaged on all three test sets, of 9.15% and 13.98% is achieved for multi- and clean-condition training respectively. In comparison with the performance achieved by the standard baseline system, our approach achieves an overall error rate reduction of 54.60% averaged for both cases.

6. REFERENCES

[1] A. Biem and S. Katagiri, “Feature extraction based on minimum classification error/generalized probabilistic descent method", *Proc. of ICASSP-1993*, 1993, pp.II-275-278.

[2] R. Chengalvarayan, “Evaluation of front-end features and noise compensation methods for robust Mandarin speech recognition", *Proc. of Eurospeech-2001*, Denmark, 2001.

[3] W. Chou et al., “Signal conditioned minimum error rate training", *Proc. of Eurospeech-1995*, 1995, pp.495-498.

[4] L. Deng, A. Acero, M. Plumpe, and X.-D. Huang, “Large-vocabulary speech recognition under adverse acoustic environments", *Proc. of ICSLP-2000*, China, October 2000.

[5] J. Droppo, L. Deng and A. Acero, “Evaluation of the SPLICE algorithm on the Aurora2 database", *Proc. of Eurospeech-2001*, Aalborg, Denmark, September 2001.

[6] H. G. Hirsch and D. Pearce, “The AURORA experimental framework for the performance evaluations of speech recognition systems under noisy conditions", *ISCA ITRW ASR2000*, Paris, France, September 2000.

[7] B.-H. Juang, W. Chou and C.-H. Lee, “Minimum classification error rate methods for speech recognition", *IEEE Trans. on Speech and Audio Processing*, Vol. 5, pp.257-265, 1997.

[8] P. Moreno, *Speech Recognition in Noisy Environments*, Ph.D. thesis, Carnegie Mellon University, 1996.

[9] K.K. Paliwal, M. Bacchini and Y. Sagisaka, “Simultaneous design of feature extractor and pattern classifier using the minimum classification error training algorithm", *Proc. of NNSP-1995*, 1995, pp.67-76.

[10] M. Rahim and C.-H. Lee, “Simultaneous feature and HMM design using string-based minimum classification error training criterion", *Proc. ICSLP-96*, 1996, pp. 1820-1823.

[11] *http://icslp2002.colorado.edu/special_sessions/aurora/references/aurora_ref10.pdf*.

[12] R.M. Stern et al., “Compensation for environmental degradation in automatic speech recognition", *Proc. ETRW on Robust Speech Recognition for Unknown Communication Channels*, Pont-a-Mousson, France, April 1997.

[13] S. Young et al., *The HTK Book (for HTK V3.0)*, July 2000.