AN ENVIRONMENT COMPENSATED MAXIMUM LIKELIHOOD TRAINING APPROACH BASED ON STOCHASTIC VECTOR MAPPING

Jian Wu, Qiang Huo and Donglai Zhu

Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong
(Email: jianwu@microsoft.com, qhuo@cs.hku.hk, dlzhu@cs.hku.hk)

ABSTRACT

Several recent approaches for robust speech recognition are developed based on the concept of stochastic vector mapping (SVM) that perform a frame-dependent bias removal to compensate for environmental variabilities in both training and recognition stages. Some of them require the stereo recordings of both clean and noisy speech for the estimation of SVM function parameters. In this paper, we present a detailed formulation of an maximum likelihood training approach for the joint design of SVM function parameters and HMM parameters of a speech recognizer that does not rely on the availability of stereo training data. Its learning behavior and effectiveness is demonstrated by using the experimental results on Aurora3 Finnish connected digits database recorded by using both close-talking and hands-free microphones in cars.

1. INTRODUCTION

Using frame-dependent bias removal to compensate for environmental variabilities has been studied in robust automatic speech recognition (ASR) area for many years (e.g. [1, 9, 3, 8]). Several recent approaches that perform a frame-dependent bias removal in both training and recognition stages are developed based on the concept of stochastic vector mapping (SVM) [4, 5, 11, 12]. The SPLICE algorithm proposed in [4, 5] requires the stereo recordings of both clean and noisy speech for the estimation of SVM function parameters. In many ASR applications, stereo data are too expensive to collect. In [11], an environment compensated minimum classification error (MCE) training approach was proposed for the joint design of SVM function parameters and HMM parameters of a recognizer that does not rely on the availability of stereo data. To initialize MCE training, an environment compensated maximum likelihood (ML) training approach was also developed, but described only briefly in [12] due to the lack of space. The effectiveness of the above approaches has been confirmed [12] in a series of benchmark test on Aurora2 connected digits database [7]. In this paper, we present the detailed formulation of the above ML training approach. Its learning behavior and effectiveness is demonstrated by using the experimental results on Aurora3 Finnish connected digits database. By doing so, readers will have a better idea on how to use our approach in their ASR applications.

The rest of the paper is organized as follows. In Section 2, we describe the SVM functions we used. In Section 3, we present the detailed formulation of our ML training approach. In Section 4, we report the experimental results, and finally we conclude the paper in Section 5.

This research was supported by grants from the RGC of the Hong Kong SAR (Project Numbers HKU7022/00E and HKU7039/02E). Jian Wu is now with Microsoft Corporation, Redmond, USA.

2. STOCHASTIC VECTOR MAPPING FUNCTIONS

Let's assume that a speech utterance corrupted by additive noise and convolutional distortion has been transformed into a sequence of feature vectors. Given a set of training data $\mathcal{Y} = \{Y_i\}_{i=1}^{I}$, where Y_i is a sequence of feature vectors of original speech, suppose that they can be partitioned into E environment classes, and the D-dimensional feature vector y under an environment class e follows the distribution of a mixture of Gaussians, $p(y|e) = \sum_{k=1}^{K} p(k|e)p(y|k,e) = \sum_{k=1}^{K} p(k|e)\mathcal{N}(y;\xi_k^{(e)}, R_k^{(e)})$, where $\mathcal{N}(\cdot;\xi,R)$ is a normal distribution with mean vector ξ and diagonal covariance matrix R. Readers are referred to [13] for the approach we used for the automatic clustering of environment conditions from training data $\mathcal{Y}$, the labelling of an utterance Y to a specific environment condition, and the estimation of the above model parameters. Given the set of Gaussian mixture models (GMM) $\{p(y|e)\}$, the task of frame-dependent SVM-based compensation is to estimate the compensated feature vector $\hat{x}$ from the original feature vector y by applying the environment-dependent transformation $\mathcal{F}(y;\Theta^{(e_y)})$, where Θ represents the trainable parameters of the transformation and e_y denotes the corresponding environment class to which y belongs. We have studied two SVM functions. The first one is borrowed from [5] and listed as follows:

$$\hat{x} \triangleq \mathcal{F}_1(y;\Theta^{(e)}) = y + \sum_{k=1}^{K} p(k|y,e) b_k^{(e)} \ , \tag{1}$$

where

$$p(k|y,e) = \frac{p(k|e)p(y|k,e)}{\sum_{j=1}^{K} p(j|e)p(y|j,e)} \ , \tag{2}$$

and $\Theta^{(e)} = \{b_k^{(e)}\}_{k=1}^{K}$. The second SVM function we used is borrowed from [4] and listed as follows:

$$\hat{x} \triangleq \mathcal{F}_2(y;\Theta^{(e)}) = y + b_k^{(e)} \ , \tag{3}$$

where $k = \arg\max_{k'=1,\ldots,K} p(k'|y,e)$ for the environment class e which y belongs to, and $\Theta^{(e)} = \{b_k^{(e)}\}_{k=1}^{K}$.

In recognition, given an unknown utterance Y, the most similar training environment class is identified first (e.g. [13]). Then, the corresponding GMM and the mapping function are used to derive a compensated version of $\hat{X}$ from Y. For the convenience of notation, we also use hereinafter $\mathcal{F}(Y;\Theta)$ to denote the compensated version of the utterance Y by transforming individual feature vector y_t as defined in previous SVM functions. After feature

compensation, $\hat{X}$ is finally recognized by an HMM-based recognizer trained as described in the following section.

3. JOINT ML TRAINING OF SVM FUNCTION PARAMETERS AND CDHMMS

3.1. Our Approach

Let's assume that each basic speech unit is modelled by a Gaussian mixture continuous density HMM (CDHMM) in our speech recognizer. Our environment compensated ML training approach is to maximize, by adjusting SVM function parameters Θ and CDHMM parameters Λ, the following likelihood function

$$\mathcal{L}(\Theta,\Lambda)=\prod_{i=1}^{I} p(\mathcal{F}(Y_i;\Theta)|\Lambda) \tag{4}$$

defined on the training set $\mathcal{Y}$. The detailed procedure to achieve the above is described as follows:

Step 1: *Initialization*

First, a set of CDHMMs with diagonal covariance matrix, $\Lambda = \{\pi_s, a_{ss'}, c_{sm}, \mu_{sm}=[\mu_{sm1},...,\mu_{smD}]^{Tr}, \Sigma_{sm}=diag\{\sigma^2_{sm1}, ..., \sigma^2_{smD}\}, s, s' = 1...S, m = 1...M\}$, are trained from multi-condition training data $\mathcal{Y}$ as initial values of HMM parameters. Initial values of the bias vectors $b_k^{(e)}=[b_{k1}^{(e)},...,b_{kD}^{(e)}]^{Tr}$ are set to be zero.

Step 2: *Estimating SVM Function Parameters* Θ

Second, given the HMM parameters Λ, for each environment class e, N_b (cf. Fig. 1) EM iterations are performed to estimate the environment dependent mapping function parameters $\bar{\Theta}^{(e)}$ to increase the likelihood function $\mathcal{L}(\Theta,\Lambda)$.

Let's consider a particular environment class e and use I_e to denote the subset of the subscript of training utterance Y_i which belongs to the environment class e. If the SVM function in Eq. (1) is used for feature compensation, the auxiliary $\mathcal{Q}$-function for $\Theta^{(e)}$ becomes

$$\begin{aligned}\mathcal{Q}_e &= \sum_{i\in I_e}\sum_t\sum_s\sum_m \zeta_{it}(s,m)\log\mathcal{N}(y_{it} \\ &\quad + \sum_k p(k|y_{it},e)b_k^{(e)};\mu_{sm},\Sigma_{sm}) \\ &= \sum_{i\in I_e}\sum_t\sum_s\sum_m \zeta_{it}(s,m)\Sigma_{sm}^{-1}(y_{it} \\ &\quad + \sum_k p(k|y_{it},e)b_k^{(e)} - \mu_{sm})^2 + \textit{Const}\ . \end{aligned} \tag{5}$$

In the above equation, $\zeta_{it}(s,m)$ is the occupation probability of Gaussian component m in state s, at time t of current compensated observation, $\hat{x}_t$. 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} is the t-th frame feature vector of utterance Y_i. *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

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

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

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

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

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

for all $k=1,...,K$.

If the SVM function in Eq. (3) is used for feature compensation, the EM updating formula for $b_k^{(e)}$ can be derived similarly with a much simpler result as follows:

$$b_{kd}^{(e)}=\frac{\sum_{i\in I_e}\sum_{t,s,m}\mathbf{1}[k,i,t]\zeta_{it}(s,m)(\mu_{smd}-y_{it}^{(d)})/\sigma^2_{smd}}{\sum_{i\in I_e}\sum_{t,s,m}\mathbf{1}[k,i,t]\zeta_{it}(s,m)/\sigma^2_{smd}}\ , \tag{10}$$

where

$$\mathbf{1}[k,i,t]=\begin{cases}1 \text{ if } k=\arg\max_{k'}p(k'|y_{it},e)\\ \\ 0 \text{ otherwise}\end{cases}\ . \tag{11}$$

It is noted that the above updating formula is similar to the bias estimation formula of feature-space stochastic matching approach reported in [10].

Step 3: *Estimating CDHMM Parameters* Λ

Third, we transform each training utterance using the relevant mapping function with parameters $\bar{\Theta}^{(e)}$. Using the environment compensated utterances, N_h (cf. Fig. 1) EM iterations are performed to re-estimate CDHMM parameters $\bar{\Lambda}$, with an increase of the likelihood function $\mathcal{L}(\bar{\Theta},\Lambda)$.

Step 4: *Repeat* **Step 2** *and* **Step 3** N_e *times* (cf. Fig. 1).

After the above steps, we obtain the $\bar{\Theta}$ and $\bar{\Lambda}$ as an ML estimation of mapping function parameters and CDHMM parameters, which can be used for recognition directly or as initial values for further joint MCE training as described in [11]. The above training procedure is illustrated in Fig. 1.

3.2. Discussion

It is noted that solving D equations in Eq. (7) involves an inverse operation of D $K\times K$ matrices. To reduce computational complexity, the following hybrid approach is actually used in our experiments when the SVM function in Eq. (1) is used for feature compensation:

- In Step 2 of the above ML training procedure, Eq. (3) is used for feature compensation and Eq. (10) is used for updating $\Theta^{(e)}$;
- In Step 3 of the above ML training procedure and in recognition stage, Eq. (1) is used for feature compensation.

Of course, another scenario is to use Eq. (3) consistently in all the steps of ML training as well as in recognition stage. In the next section, we report experimental results of the above two approaches which are referred as SVM1 and SVM2 respectively.

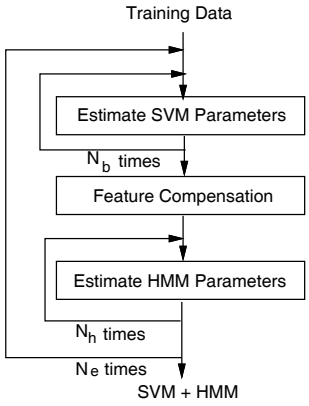


Fig. 1. An illustration of our joint ML training procedure.

4. EXPERIMENTAL RESULTS

4.1. Experimental Setup

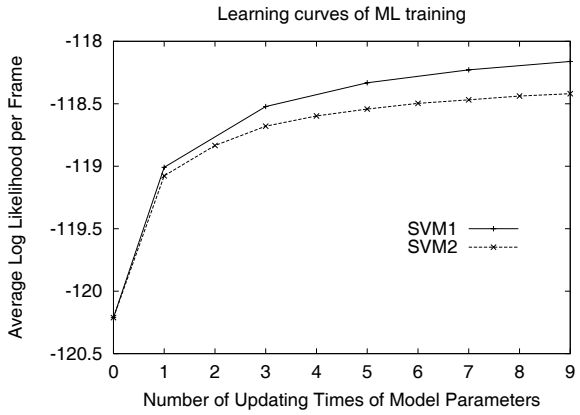
We use Aurora3 database to verify our algorithm. Aurora3 contains utterances of connected digits in four European languages, namely Finnish, Spanish, German and Danish. All utterances were recorded by using both close-talking (CT) and hands-free (HF) microphones in cars under several driving conditions to reflect some realistic scenarios for typical in-vehicle ASR applications. For each language, the database is divided into three subsets according to matching degree between training data and test data: Well-Matched condition (WM), Middle Mismatched condition (MM) and High Mismatched condition (HM). In the following discussions, only the results on Finnish Database [2] under well-matched condition are used.

The front-end used in our experiments is the ETSI Advanced Front-End (AFE) as described in [6]. A feature vector sequence is extracted from the input speech utterance via a sequence of processing modules that include noise reduction, waveform processing, cepstrum calculation, blind equalization, and "server feature processing". Each frame of feature vector has 39 features that consists of 12 MFCCs (C_1 to C_{12}), a combined log energy and C_0 term, and their first and second order derivatives. All the feature vectors are computed from a given speech utterance, but the feature vectors that are sent to the speech recognizer and the training module are those corresponding to speech frames, as detected by a VAD module described in Annex A of [6].

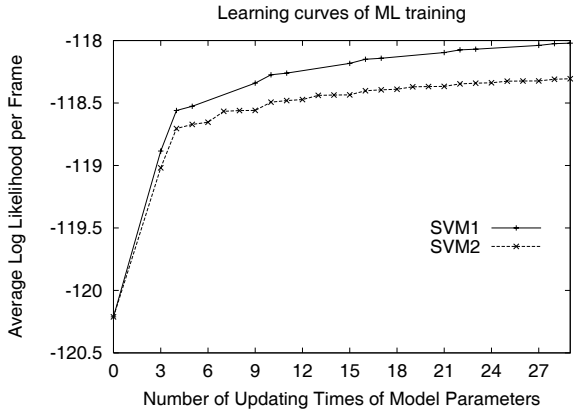
Each digit is modelled as a whole word left-to-right CDHMM with 16 emitting states, 3 Gaussian mixture components with diagonal covariance matrices per state. Besides, two pause models, "sil" and "sp", are created to model the silence before/after the digit string and the short pause between any two digits. The "sil" model is a 3-emitting state CDHMM with a flexible transition structure as that of HMM described in [7]. Each state is modelled by a mixture of 6 Gaussian components with diagonal covariance matrices. The "sp" model consists of 2 dummy states and a single emitting state which is tied with the middle state of "sil".

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

Two sets of CDHMMs are trained by using all training data



(a) $N_b = 1, N_h = 1, N_e = 5$



(b) $N_b = 3, N_h = 3, N_e = 5$

Fig. 2. Learning behavior of two ML training approaches under two experimental settings.

without SVM-based feature compensation under ML and MCE criterion respectively, where the MCE-training is initialized by the ML-trained models. The performance (word error rate in %) of these two baseline systems are 3.95% and 2.82%, respectively.

In SVM experiments, all the training data are clustered into 8 different environment classes, of which each is modeled by a GMM consisting of 32 Gaussian components [13]. Therefore, there are $8 \times 32 = 256$ bias vectors to be estimated as the mapping function parameters.

4.2. A Comparison of Learning Behavior in Likelihood

To understand the learning behavior of two ML training approaches, i.e., SVM1 and SVM2, their learning curves under two settings, $N_b = 1, N_h = 1, N_e = 5$ and $N_b = 3, N_h = 3, N_e = 5$, are illustrated in Fig. 2 (a) and (b) respectively. In the above curves, the horizontal axis is the cumulative number of updating times of either SVM or HMM parameters, and the vertical axis is the average log likelihood per frame, i.e. $\frac{log\mathcal{L}(\Theta,\Lambda)}{Total\ number\ of\ frames}$. SVM2 approach can guarantee a monotonic increase of its likelihood function, while SVM1 approach cannot because a hybrid approach is used in our implementation. For SVM1 curves, we only include those points with likelihood values obtained in Step 3 of the ML training procedure because the likelihood function

Table 1. Word error rate (in %) of ML/MCE-trained models based on stochastic vector mapping with different mapping functions (SVM1 *vs* SVM2) and iteration numbers (N_b, N_h, N_e) as defined in Fig. 1 under WM condition of Finnish language.

Mapping Function (Iteration Number)	ML	MCE
Baseline (without SVM)	3.95	2.82
SVM1 ($N_b = 1, N_h = 1, N_e = 1$)	3.49	2.36
SVM2 ($N_b = 1, N_h = 1, N_e = 1$)	3.51	2.57
SVM1 ($N_b = 1, N_h = 1, N_e = 5$)	3.62	2.60
SVM2 ($N_b = 1, N_h = 1, N_e = 5$)	3.68	2.74
SVM1 ($N_b = 1, N_h = 5, N_e = 1$)	**3.33**	**2.34**
SVM2 ($N_b = 1, N_h = 5, N_e = 1$)	**3.34**	**2.60**
SVM1 ($N_b = 3, N_h = 3, N_e = 5$)	3.65	2.65
SVM2 ($N_b = 3, N_h = 3, N_e = 5$)	3.68	2.68

in this step is consistent with the original one. Apparently, both approaches have a good learning behavior in terms of increasing the likelihood functions. Note that different likelihood functions are used in SVM1 and SVM2 approaches because different SVM functions are used for feature compensation.

4.3. A Comparison of Recognition Performance

Different choices of mapping functions and iteration numbers will lead to different results of parameter estimation. Such differences will be reflected not only in the likelihood value, but more importantly, in the recognition performance of final ML-trained models. Furthermore, as suggested in [12], the environment compensated MCE training approach will use ML-trained models as seed models for joint MCE training. Therefore, different ML-trained seed models will result in different recognition performance of MCE-trained models. In order to compare recognition performance with different setup of ML training procedure and to identify a "good" setting of relevant control parameters, a series of experiments are conducted and their results are summarized in Table 1.

By comparing each pair of rows for SVM1 and SVM2 with the same setting for (N_b, N_h, N_e), it is observed that no big difference is made by different forms of SVM1 and SVM2 in terms of the final recognition performance for ML-training. This observation is consistent with the one in the study of SPLICE approach [5]. However, for MCE-training, SVM1 performs better than SVM2. By comparing those rows for SVM1 or SVM2 with different settings for (N_b, N_h, N_e), it is observed that the setting $N_b = 1, N_h = 5, N_e = 1$ gives the best recognition performance. A detailed analysis of our results reveals that multiple iterations for updating SVM function parameters in Step 2 of our ML training procedure is harmful in terms of improving recognition performance, while multiple iterations for updating HMM parameters in Step 3 is helpful. In comparison with the ML/MCE-trained baseline systems without using SVM for feature compensation, the two best SVM-based systems can reduce the word error rates by 15.7% and 17% for ML and MCE training respectively.

5. CONCLUSIONS AND DISCUSSIONS

In this paper, we have presented a detailed formulation of an environment compensated ML training approach for the joint design of the SVM-based feature compensation module and HMM parameters of a speech recognizer. For readers who have interest to try our approach, our recommendations are: (1) You can take either form of SVM functions we studied here; (2) Set $N_b = 1$, $N_e = 1$ and run several iterations (i.e. N_h) for training HMM parameters.

Although we have demonstrated the usefulness of the SVM-based approaches for several robust ASR applications where diversified yet representative training data are available, the performance improvement of SVM-based approaches is less significant in the case of that there is a severe mismatch between training and testing conditions. In order to improve the performance further, one possibility is to perform unsupervised online adaptation of SVM function parameters. We have conducted a study along this direction and will report its results elsewhere.

6. REFERENCES

[1] A. Acero, *Acoustic and Environment Robustness in Automatic Speech Recognition*, Kluwer Academic Publishers, 1993.

[2] Aurora document AU/217/99, "Availability of Finnish SpeechDat-Car database for ETSI STQ WI008 front-end standardisation," Nokia, Nov 1999.

[3] W. Chou, M. G. Rahim, and E. Buhrke, "Signal conditioned minimum error rate training," *Proc. 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. ICSLP-2000*, Oct. 2000.

[5] L. Deng, A. Acero, L. Jiang, J. Droppo, and X.-D. Huang, "High-performance robust speech recognition using stereo training data," *Proc. ICASSP-2001*, 2001, pp.I-301-304.

[6] ETSI standard document, "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), 2002.

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

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

[9] M. Rahim, B.-H. Juang, W. Chou, and E. Buhrke, "Signal conditioning techniques for robust speech recognition," *IEEE Signal Processing Letters*, Vol. 3, No. 4, pp.107-109, 1996.

[10] A. Sankar and C.-H. Lee, "A maximum-likelihood approach to stochastic matching for robust speech recognition," *IEEE Trans. on Speech and Audio Processing*, Vol. 4, No. 3, pp.190-202, 1996.

[11] J. Wu and Q. Huo, "An environment compensated minimum classification error training approach and its evaluation on Aurora2 database," *Proc. ICSLP-2002*, Denver, 2002, pp.I-453-456.

[12] J. Wu and Q. Huo, "Several HKU approaches for robust speech recognition and their evaluation on Aurora connected digit recognition tasks," *Proc. Eurospeech-2003*, 2003, pp.21-24.

[13] J. Wu, D. Zhu and Q. Huo, "A study of minimum classification error training for segmental switching linear Gaussian hidden Markov models," *Proc. ICSLP-2004*, 2004.