ROBUST SPEECH RECOGNITION USING DISCRIMINATIVE STREAM WEIGHTING AND PARAMETER INTERPOLATION

Stephen M. Chu and Yunxin Zhao

Beckman Institute and Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

ABSTRACT

This paper presents a method to improve the robustness of speech recognition in noisy conditions. It has been shown that using dynamic features in addition to static features can improve the noise robustness of speech recognizers. In this work we show that in a continuous-density Hidden Markov Model (HMM) based speech recognition system, weighting the contribution of the dynamic features according to SNR levels can further improve the performance, and we propose a two-step scheme to adapt the weights for a given Signal to Noise Ratio (SNR). The first step is to obtain the optimal weights for a set of selected SNR levels by discriminative training. The Generalized Probabilistic Decent (GPD) framework is used in our experiments. The second step is to interpolate the set of SNR-specific weights obtained in step one for a new SNR condition. Experimental results obtained by the proposed technique is encouraging. Evaluation using speaker independent digits with added white Gaussian noise shows significant reduction in error rate at various SNR levels.

1. INTRODUCTION

HMM based speech recognizers perform well if trained and tested under matched conditions. However, trying to train models for all possible environments and SNR levels is impractical. Therefore one way to solve this problem is to compensate models trained on clean speech to give adapted models that are more robust to noise. It is desired that the compensation method be computational inexpensive so that the models can be quickly adjusted to suit the noise conditions.

It is well known that using dynamic features in addition to static features can improve the noise robustness of speech recognizers. In this work we demonstrate that weighting the contribution of the dynamic features according to SNR levels can further improve the performance in multi-stream HMM speech recognition systems.

In such a system, the static and dynamic features within each HMM are modeled by two separate streams. For each stream in each state of each model, an exponential weight is used to control this particular stream's contribution to the output probability. The objective is to adjust these stream weights so that the recognition error is minimized. Recently many researchers have shown that discriminative training methods, especially the GPD algorithm, are effective in finding the stream weights that minimize the recognition error [2]. However, majority of the reported experiments are carried out using clean speech. In our experiment, we apply the discriminative training algorithm under noisy conditions to gain insight of the relationship between the optimal stream weights and SNR. Moreover, since each weight can be treated as a function of SNR, we are able to obtain the weights for an unseen SNR by interpolating the known points. During recognition, the SNR of the noisy speech signal is first estimated. The stream weights at that SNR are then calculated and updated. It is worthy noting that the calculation of weights at the recognition stage is very simple and straightforward.

This paper is organized as follows. In the next section we introduce the multi-stream HMM and the exponent stream weights. We start section 3 by reviewing the GPD formulation, which is then followed by the derivation of the formulas used in the discriminative training of the weights. The two-step parameter interpolation scheme is discussed in section 4. We present the experimental results in section 5 and conclude the work in section 6.

2. STREAM WEIGHTING IN MULTI-STREAM HMM

Let's first consider a typical single-stream continuous-density HMM. The output distribution $b_{i,j}(o_t)$ of model *i*, state *j*, given the observation vector at time *t*, is represented by a mixture of Gaussian densities. The formula for computing $b_{i,j}(o_t)$ is

$$b_{i,j}(o_t) = \sum_{m=1}^{M} c_{i,j,m} N(o_t, \mu_{i,j,m}, \Sigma_{i,j,m}) \qquad (1)$$

where M is the number of mixture components, $c_{i,j,m}$ is the weight of the m'th component and $N(x,\mu_{i,j,m},\Sigma_{i,j,m})$ is a multivariate Gaussian with mean vector $\mu_{i,j,m}$ and covariance matrix $\Sigma_{i,j,m}$. The mixture weights $c_{i,j,m}$ in the above equation are nonnegative and add up to one.

A single-stream HMM can be used to model both static and dynamic features of speech if we concatenate the static and dynamic features to form a single observation vector.

However, if we assume that the static features and the dynamic features are statistically independent, then we can put them into two separate streams. Let's denote the static observation vector at time *t* by $o_{1,t}$ and the dynamic observation vector by $o_{2,t}$, the formula for computing $b_{i,j}(o_t)$ is then

$$b_{i,j}(o_t) = \prod_{s=1}^{2}\left[\sum_{m=1}^{M} c_{i,j,s,m} N(o_{s,t}, \mu_{i,j,s,m}, \Sigma_{i,j,s,m})\right] \qquad (2)$$

Putting static and dynamic features into separate streams allows us to introduce exponent stream weights to control each stream's contribution to the final output probability, and thus weighting the "importance" of each feature stream in the model. The formula for computing the output probability using the exponent stream weights is

$$b_{i,j}(o_t) = \prod_{s=1}^{2}\left[\sum_{m=1}^{M} c_{i,j,s,m} N(o_{s,t}, \mu_{i,j,s,m}, \Sigma_{i,j,s,m})\right]^{\gamma_{i,j,s}} \qquad (3)$$

where $\gamma_{i,j,s}$ is the weight of stream s of state j in model i. Notice that since the new $b_{i,j}(o_t)$ does not satisfy stochastic constrains in general, equation (3) is no longer a probability mass function.

The exponent stream weights $\gamma_{i,j,s}$ in equation (3) are model and state specific, i.e. for each state in each model, there is a specific set of stream weights. However, the weights can also be tied at various levels. Two possible ways of tying are tying at the global level and tying at the class level. In the first case, all states from all models share the same set of stream weights, and in the second case, all states in the same HMM have the same stream weights.

The main focus of the current work is to improve the noise robustness of automatic speech recognition by finding the optimal stream weights at a given SNR level. In the training stage, we use discriminative methods to adjust the weights to achieve minimum recognition rate. We discuss the discriminative training in the next section.

3. DISCRIMINATIVE TRAINING

3.1. Minimum Error Classification

Discriminative training for minimum classification error attempts to minimize a function of recognition errors on a closed data set. The most important aspect of minimum error classification is the definition of a suitable objective function, the minimization of which is directly related to the minimization of the empirical error rate. The objective function should be a smooth differentiable function of the HMM parameters (the exponent stream weights in our case), with an easily computable first order derivative, so that the gradient descent algorithms can be used to minimize the recognition error.

The objective function is derived by the three-step procedure prescribed in the GPD formulation [1].

3.2. GPD Formulation

The first step of the formulation is to define an appropriate discriminant function $g_i(\mathbf{x};\Lambda)$ which is used to implement the following decision rule

$$C(\mathbf{x}) = C^i \quad \text{if} \quad g_i(\mathbf{x};\Lambda) = \max_j g_j(\mathbf{x};\Lambda) \qquad (4)$$

where $C(\mathbf{x})$ denotes the recognition decisions on **x**, C^i denotes class i, and Λ denotes the HMM parameter set including the stream weights. The discriminative function $g_i(\mathbf{x};\Lambda)$ is defined as

$$g_i(\mathbf{x};\Lambda) = \ln\left(\left[\sum_{\text{all }\Theta_i}\left\{P\left(\mathbf{x},\Theta_i|\Lambda\right)\right\}^{\xi}\right]^{1/\xi}\right) \qquad (5)$$

where Θ_i is any allowable state sequence for the i'th HMM class, i.e.

$$\Theta_i = \{\theta_{i,1},\ldots,\theta_{i,t}\ldots\} \qquad (6)$$

When $\xi = 1$, equation (5) becomes the standard HMM log likelihood probability, which is usually computed using the forward-backward algorithm. When $\xi = \infty$, equation (5) reduces to the log likelihood of the Viterbi best path. Let's now redefine Θ as the Viterbi state sequence for the corresponding HMM,

$$\Theta = \{\theta_1,\theta_2,\ldots,\theta_t,\ldots\} \qquad (7)$$

then $g_i(\mathbf{x};\Lambda)$ is simplified to

$$g_i(\mathbf{x};\Lambda) = \ln\left(\pi_{i,\theta_1}\right) + \sum_{t=1}^{T_x}\ln\left(a_{i,\theta_{t-1},\theta_t}\right) + \sum_{t=1}^{T_x}\ln\left(b_{i,\theta_t}(o_t)\right) \qquad (8)$$

where o_t is the feature vector of observation **x** at time t, π_{i,θ_1} is the initial probability of state θ_1 in the i'th HMM, $a_{i,\theta_{t-1},\theta_t}$ is the state transition probability from state θ_{t-1} to θ_t in the i'th HMM, $b_{i,\theta_t}(o_t)$ is the output probability of state θ_t for o_t, and $T_{\mathbf{x}}$ is the total number of feature vectors in the observation **x**.

The second step of GPD formulation is to introduce a misclassification measure in order to embed the decision process in a function form. The misclassification measure for class i is defined by

$$d_i(\mathbf{x};\Lambda) = -g_i(\mathbf{x};\Lambda) + \ln\left(\frac{1}{M-1}\sum_{j,j\neq i}\exp\left[\eta g_j(\mathbf{x};\Lambda)\right]\right)^{1/\eta} \qquad (9)$$

where η is a positive number. $d_i(\mathbf{x};\Lambda)$ calculates the penalty incurred when classifying the current observation **x** as the i'th class.

The third step is to define a smooth loss function $l_i(\mathbf{x};\Lambda)$ for each class based on the misclassification measure. We choose the sigmoid function in our implementation. Therefore

$$l_i(\mathbf{x};\Lambda) = \frac{1}{1+\exp\left(-\alpha\left[d_i(\mathbf{x};\Lambda)-\beta\right]\right)} \qquad (10)$$

The constant α and β control the slope and mid-point of the sigmoid. Both α and β are positive real numbers and are determined experimentally.

3.3. Adaptive Learning

If **x** is a labeled training sample of class *i*, then the loss incurred when this sample is presented is $l_i(\mathbf{x};\Lambda)$. This is the objective function used in our experiments. The parameter set Λ is adjusted recursively when each training sample is presented. Let $\mathbf{x}_n$ be the *n*'th training sample and Λ_{n+1} be the parameter set after $\mathbf{x}_n$ is applied. Then Λ_{n+1} can be calculated by the following rule

$$\Lambda_{n+1} = \Lambda_n + \Delta\Lambda_n \tag{11}$$

where

$$\Delta\Lambda_n = -\varepsilon(n)\mathbf{U}\nabla l(\mathbf{x}_n;\Lambda_n) \tag{12}$$

In equation (12), **U** is a positive definite matrix, ∇ is the gradient operator, and $\varepsilon(n)$ is a monotonic decreasing function of *n* that gives a small positive learning rate. In this experiment we use

$$\varepsilon(n) = \frac{a}{b+nc} \tag{13}$$

where *a*, *b*, and *c* are experimentally determined positive constants. Assuming **U** to be the identity matrix, the adjusting formula (11) for the exponent weights of equation (3) becomes

$$\gamma_{i,j,s}^{n+1} = \gamma_{i,j,s}^{n} - \varepsilon(n)\frac{\partial l_i(\mathbf{x}_n;\Lambda_n)}{\partial \gamma_{i,j,s}^{n}}, \qquad \text{if } C(\mathbf{x}_n) = C^i \tag{14}$$

One can also use the empirical loss as the objective function, which is calculated by summing up $l_i(\mathbf{x};\Lambda)$ over all training samples of all classes. In this case, the parameters are updated after all samples in the training set is presented.

3.4. Differentiate the Loss Function

To realize the adaptive learning algorithm described in the previous section, it is necessary to calculate the derivatives of the loss function with respect to the exponent stream weights. The computation of the derivatives is accomplished by applying the chain rule, i.e.,

$$\frac{\partial l_i(\mathbf{x};\Lambda)}{\partial \gamma_{i,j,s}} = \frac{\mathrm{d}\, l_i\left(d_i(\mathbf{x};\Lambda)\right)}{\mathrm{d}\, d_i(\mathbf{x};\Lambda)} \cdot \frac{\partial d_i(\mathbf{x};\Lambda)}{\partial \gamma_{i,j,s}} \tag{15}$$

The first term in the right hand side of equation (15) can be evaluated as

$$\frac{\mathrm{d}\, l_i\left(d_i(\mathbf{x};\Lambda)\right)}{\mathrm{d}\, d_i(\mathbf{x};\Lambda)} = \frac{\alpha l_i\left(d_i(\mathbf{x};\Lambda)\right)^2}{\exp\left(\alpha\left(d_i(\mathbf{x};\Lambda) - \beta\right)\right)} \tag{16}$$

Assuming the exponent stream weights to be model and state specific, and upon differentiating the second term, we get

$$\begin{aligned}\frac{\partial d_i(\mathbf{x};\Lambda)}{\partial \gamma_{i,j,s}} &= -\frac{\partial g_i(\mathbf{x};\Lambda)}{\partial \gamma_{i,j,s}} \\ &= -\sum_{t=1}^{T_x} \delta(\theta_t - j) \cdot \frac{\partial \ln\left(b_{i,j}(o_t)\right)}{\partial \gamma_{i,j,s}}\end{aligned} \tag{17}$$

Using equation (3), we can obtain

$$\frac{\partial \ln\left(b_{i,j}(o_t)\right)}{\partial \gamma_{i,j,s}} = \ln\left[\sum_{m=1}^{M} c_{i,j,s,m} N\left(o_{s,t}, \mu_{i,j,s,m}, \Sigma_{i,j,s,m}\right)\right] \tag{18}$$

Now, given a labeled training sample **x**, we can readily use equation (14) to update the corresponding stream weights.

4. PARAMETER INTERPOLATION

The parameter interpolation procedure involves two steps for a given type of noise corruption.

In the first step, we obtain the optimal stream weights at a set of known SNR levels. These SNR settings should cover a wide range of noise levels so that interpolation in the following step is feasible. We start with a set of multi-stream HMMs that are trained using clean speech. At each SNR setting, the stream weights in these HMMs are adjusted to achieve minimum recognition error by the discriminative training algorithm discussed in the previous section using noisy speech with corresponding SNR. This step is done in the training stage.

The second step is carried out in the recognition stage when a test utterance with an unseen SNR is presented. The goal of the interpolation is to calculate the stream weights that can reduce the recognition error without having to go through the training stage for all SNRs. Let *m* be the total number of the stream weights. Then the interpolation process for a new SNR can be viewed as finding the trajectory along the existing SNR s for the *m*-dimensional stream weights. A simpler alternative is to assume the weights are independent to each other. And treat each of them as a separate function of SNR. Thus the interpolation becomes *m* independent 1-D interpolations. Linear interpolation using the exponent weights of the two neighboring SNR conditions is used in this experiment for its simplicity.

5. EXPERIMENTS

5.1. Experimental Paradigm

The experiments were conducted using the speaker independent isolated digit TI database. The data was pre-processed using a 25 ms Hamming window and has a 10 ms frame shift. For each frame, a set of 15 MFCC coefficients with an analysis order of 24 were computed. The delta coefficients were also calculated. For each digit, a HMM with 8 emitting states and Gaussian density for each state was trained by EM method using clean speech data. Diagonal covariance matrices were used in the HMMs.

The discriminative training programs were built using the HTK library functions. The stream weights for six SNR levels of 0dB, 10dB, 20dB, 30dB, 40dB, and 50dB were discriminatively adjusted to minimize recognition error at the given SNR. The stream weights had been initialized to be $\gamma_{i,j,s}\equiv1$ for all i, j, s prior to the discriminative training. The noisy speech samples were synthesized by adding white Gaussian noise to the clean speech.

To evaluate the proposed parameter interpolation scheme, 5 test data sets with unseen SNR levels were evaluated using the stream weights obtained from interpolation. The results are shown and discussed next.

5.2. Results

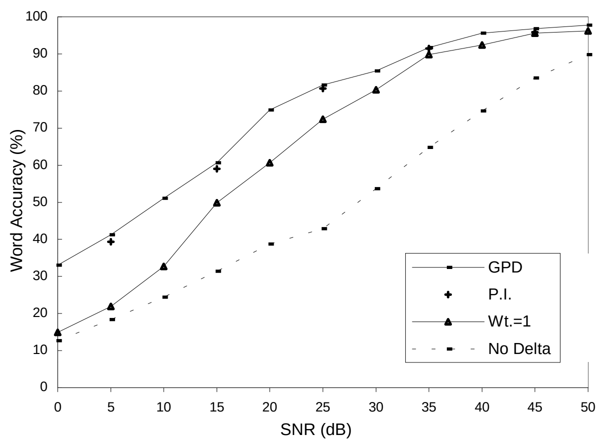

Figure 1. Comparing recognition performance at different stream weight configurations. (GPD: use stream weights obtained from GPD training; P.I.: use stream weights obtained from parameter interpolation; Wt.=1: all weights are set to one; No Delta: no dynamic features used.)

From Figure 1. We can see that incorporating dynamic features improves the noise robustness of the recognition system. Furthermore, using stream weights obtained from discriminative training enhances the recognition performance considerably at all SNR levels. To test the interpolation scheme, five SNR settings which is not pre-trained, namely, 5 dB, 15 dB, 25 dB, 35 dB, and 45 dB, were tested using the stream weights predicted by interpolations. In order to measure the effectiveness of the interpolation method, GPD-trained weights are also evaluated at these SNR levels, as they should give an upper limit of the performance we expect from the proposed technique. Clearly, the parameter interpolation works very well. The recognition accuracy using the stream weights predicted by interpolations is only slightly lower than using the GPD-trained weights at the same SNR. The corresponding recognition accuracy readings are summarized in Table 1.

SNR (dB)	No Delta	Wt.=1	GPD.	P.I..
5	18.4	21.9	41.3	39.4
15	31.4	49.8	60.6	59.1
25	42.9	72.4	81.6	80.6
35	64.8	89.8	91.8	91.4
45	83.5	95.6	96.8	95.9

Table 1. Summary of recognition accuracy (%)

6. CONCLUSIONS

We considered the problem of HMM stream weighting in the context of robust noisy speech recognition. We proposed a two-step parameter interpolation scheme to adapt the weights for a given SNR. The GPD algorithm is used to accomplish the discriminative adaptation of the exponent stream weights during training. Using linear interpolation to predict stream weights for untrained SNR in the testing stage yields satisfactory results.

7. ACKNOWLEDGMENTS

We acknowledge the support from the Army Research Laboratory. Thanks must also be given to Prof. Thomas Huang for his stimulating discussions.

8. REFERENCES.

1. Juang, B.-H., and Katagiri, S. “Discriminative Learning for Minimum Error Classification,” *IEEE Trans. On Signal Processing*, Vol. 40: 3043-3054, 1992.

2. Hernando, J., Ayarte J., and Monte E. “Optimization of Speech Parameter Weighting for CDHMM Word Recognition,” *Proc. Eurospeech95*, Madrid, Vol. 1: 105-108, 1995.

3. Hernando, J., “Maximum Likelihood Weighting of Dynamic Speech Features for CDHMM Speech Recognition,” *Proc. ICCASP97*, Munich, Vol. 2: 1267-1270, 1997

4. Rainton, D., and Sagayama, S., “Minimum Error Classification Training of HMMs-Implementation Details and Experimental Results,” *J. Acoust. Soc. Jpn.*, Vol. 13: 379-387, 1992.

5. Chou, W., Juang, B.-H., and Lee, C.H., “Segmental GPD Training of HMM Based Speech Recognizer,” *Proc. ICCASP92*, San Francisco, Vol. 1: 473-476, 1992.

6. Rabinar, L., and Juang, B.-H., *Fundamentals of Speech Recognition*, Prentice Hall, Englewood Cliffs, 1993.