OPTIMIZATION OF TIME-FREQUENCY MASKING FILTERS USING THE MINIMUM CLASSIFICATION ERROR CRITERION

Michiel Bacchiani [1] *Kiyoaki Aikawa* [1]

[1] ATR Human Information Processing Research Laboratories
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02 Japan

ABSTRACT

The dynamic cepstrum parameter representing a masked spectrum performed extremely well in continuous speech recognition. This paper proposes a new algorithm for optimizing the dynamic cepstrum lifter array. The masking filter is represented by a set of Gaussian-shaped lifters. The standard deviation and the gain of the Gaussians are trained in order to improve the performance of the time-frequency filter. Parameterizing the lifter shape provides robustness against unknown speech samples. Because of the parameterized lifter's small degree of freedom, it can avoid overlearning. The gradient descent optimizing algorithm is formulated for both a Neural Network classifier and an HMM classifier. The optimized dynamic cepstrum successfully improved the speech recognition performance for the speech spoken even in a different speaking style.

1. INTRODUCTION

Previous research showed that improved recognition results can be obtained by using the dynamic cepstrum [1]. This representation simulates the time-frequency response of the forward masking, which is characterized by the phenomenon that the masking pattern is gradually smoothed on the frequency axis as the masker-signal interval increases. One reason for this excellent speech recognition performance is that the dynamic cepstrum removes the global spectral shape, which is dependent on the speaker identity, and also enhances the spectral dynamics, which contain important information for phoneme recognition.

This dynamic cepstrum is obtained from the cepstral representation of speech data by liftering the cepstral time-sequence using a lifter array. Several types of lifter arrays were used in previous experiments, but no other optimization techniques other than heuristic selection of the parameter combination were attempted. Even though the lifter array is represented by only a few parameters, it is difficult to achieve a true optimal combination of parameters by the heuristic approach. An optimization algorithm was desired to improve the speech recognition performance of the dynamic cepstrum.

Optimization of a single cepstral lifter using a neural network approach has been reported [2]. By extending this, a lifter array can be implemented in a Neural Network and trained using the backpropagation algorithm. However, it is likely to lose the robustness against unknown speech tokens due to the large degree of freedom.

This paper describes an optimization method for a *parameterized* lifter array using the Minimum Classification Error (MCE) criterion. The lifter shape is a Gaussian and is represented by only the standard deviation and the gain. The advantage of using a parameterized set of lifters is that the liftering array can be represented by a much smaller number of values than the non-parameterized case requires. Parameterization increases the probability of finding an array that is robust to unknown speech, because the lifters in the array are prevented from overlearning. Algorithms that facilitate supervised learning of the array parameters are proposed in this paper.

2. DYNAMIC CEPSTRUM

The dynamic cepstrum is the inverse Fourier transform of a masked spectrum and can be calculated by using the following relationship:

$$b_k(i) = \sum_{n=0}^{N} c_k(i-n) l_k(n) \qquad (1)$$

$$\text{with} \quad \begin{array}{ll} l_k(n) = 1 & n = 0 \\ l_k(n) \leq 0 & n \geq 1 \end{array}$$

where $l_k(n)$ is a low-quefrency-pass lifter, c_k is kth-order cepstral coefficient, and n is the time delay. N denotes the duration of the masking effect. This relationship reveals that the dynamic cepstrum is obtained by subtracting the weighted sum of the smoothed versions of previous spectra from the present spectrum. The parameterized low-quefrency-pass lifters used to calculate the dynamic cepstrum with Eq. (1) were Gaussian-shaped and given by

$$l_k(n) = G(n) \exp\left(-\frac{k^2}{2\sigma^2(n)}\right) \qquad (2)$$

Each lifter in the array is represented by two parameters, a gain $G(n)$ and a standard deviation $\sigma(n)$. This paper proposes a new method for optimizing the parameterized lifter array $l_k(n)$ based on the MCE criterion.

3. TRAINING MODEL PARAMETERS

To get an indication of how well the dynamic cepstrum lifter array performs, an loss function is calculated based on the output of the back end classifier, which receives the dynamic cepstrum parameter as the input. To facilitate supervised learning for optimization of the lifter parameters, it is necessary to calculate the derivative of this loss function with respect to the lifter parameters. When a lifter parameter is denoted as ψ and the loss function as L, minimization of the loss function for this parameter can be performed using the gradient descent algorithm:

$$\Delta\psi = -\eta \frac{\partial L}{\partial \psi} \qquad (3)$$

0-7803-1775-0/94 $3.00

where $\Delta\psi$ denotes the change in ψ and η denotes the learning rate.

4. MCE IN A TDNN FRAMEWORK

The lifter array can be implemented in a TDNN [3]. Because the TDNN structure is well suited for implementing the filter for cepstrum time-sequences, the lifter array and classifier can both be implemented in one network. This implementation uses a four-layered TDNN, with the lifter mechanism implemented in the input layer and first hidden layer. The second hidden layer and output layer carry out the recognition task. This network architecture is depicted in Figure 1. The input layer provides the network with 15 frames of 16-dimensional cepstrum vectors. The output of the units in the first hidden layer is the dynamic cepstrum representation of the applied sample.

Note that all the connections between the input layer and the first hidden layer are defined by only eight parameters, because the connections from the units in an input layer column are bound by the lifter's parameterized shape. Moreover, because of the tied structure, all the connections are horizontally identical. The connection weights of the recognizer as well as the parameters of the feature extractor can be trained using the error backpropagation algorithm, [4]. The derivative of the squared error at the output of the recognizer with respect to the connection weights in the feature extractor can be extended to the feature extractor's parameters by the relation

$$\frac{\partial L_r}{\partial \psi} = \frac{\partial L_r}{\partial w_{j,k}} \frac{\partial w_{j,k}}{\partial \psi} \tag{4}$$

where L_r is the squared output error of the recognizer for the r-th sample and ψ one of the lifter parameters. $w_{j,k}$ is the connection weight from unit j to unit k. Each lifter is described by two parameters, and the derivative of the output error to these two parameters can be calculated by,

$$\frac{\partial L_r}{\partial G(n)} = \delta_p c_k \exp\left(-\frac{k^2}{2\sigma^2(n)}\right) \tag{5}$$

$$\frac{\partial L_r}{\partial \sigma(n)} = \delta_p c_k \frac{G(n)k^2}{\sigma^3(n)} \exp\left(-\frac{k^2}{2\sigma^2(n)}\right) \tag{6}$$

where n is the time delay modeled by this weight and c_k is the cepstral coefficient of order k.

In Figure 1 the output function of the first hidden layer is linear. The δ term for the units in the first hidden layer are calculated using,

$$\delta_p = \sum_h \delta_h w_{p,h} \tag{7}$$

where h scans all the nodes in the second hidden layer, p is the index of the unit in the first hidden layer. The calculated derivatives of the error with respect to the weights between the units in the input layer and the units in the first hidden layer can be used to calculate the derivative of a certain lifter parameter, using the chain rule. The parameters of the lifters can now be adapted by a gradient descent algorithm according to Eq. (3).

The approach chosen here consists of two steps: (1) train the recognizer with fixed feature extractor parameters until it is able to correctly classify most of the training data based on the dynamic cepstrum data, (2) adapt the feature extractor parameters without allowing any further adaptation of the recognizer.

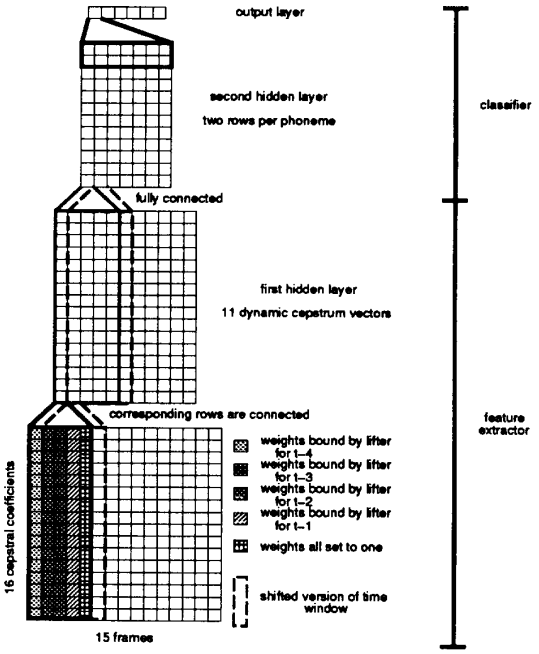

Figure 1. Integration of the feature extractor and classifier in a TDNN.

5. MCE IN AN HMM FRAMEWORK

Another MCE implementation using a set of HMMs as the back end classifier is formulated. In this implementation, diagonal mixture Gaussian HMMs are used to obtain recognition results. It is therefore necessary to define a function that is a direct measure of the number of classification errors made by the classifier. Since a gradient descent minimization of this function is requested, a differentiable loss function is needed. A method for training HMM parameters with an MCE framework has been reported [5]. This approach is applicable to the optimization of the dynamic cepstrum lifter array.

The loss function used in this paper is

$$L(\mathcal{O}, \zeta) = \sum_{r=1}^{R} \Gamma(d(\mathcal{O}_r, \zeta)) \tag{8}$$

where $\mathcal{O}$ is the observation sequence, ζ is the set of parameters, R is the number of training samples. $\Gamma(.)$ is the sigmoid function. $d(\mathcal{O}_r, \zeta)$ is a classification quality measure defined by,

$$d(\mathcal{O}_r, \zeta) = -g(\mathcal{O}_r, \zeta_c) + g(\mathcal{O}_r, \zeta_p), \tag{9}$$

ζ_c being the set of parameters of the model of the class of the sample and ζ_p the HMM parameters of the model not equal to the model of the class of the sample with the highest output probability. The $g(.)$ function is called the "discriminant function" and is defined as,

$$g(\mathcal{O}_r, \zeta_q) = \ln(P(\mathcal{O}_r \mid \zeta_q)) \tag{10}$$

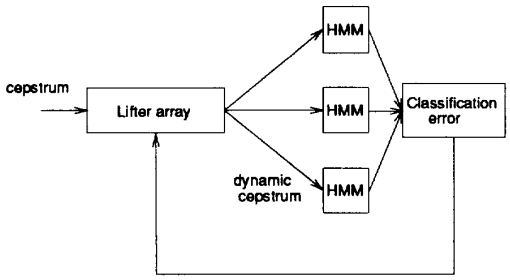

Figure 2. Schematic representation of the training method of the lifter array, using HMM's as the back end classifier.

The output probability $P(\mathcal{O}_r \mid \zeta_q)$ of the model with parameters ζ_q is defined as,

$$P(\mathcal{O}_r \mid \zeta_q) = \prod_{t \in T} a_{s_{t-1},s_t} P_{s_t}(\mathbf{o}_{r,t}) \tag{11}$$

where s_t is the state at time t along the Viterbi path. The probability $P_{s_t}(\mathbf{o}_{r,t})$ is the probability of the observation vector of sample r at time t in the state s_t, and given by the mixture of Gaussian probability distributions. a is the state-transition probability.

The use of the Viterbi alignment HMM is convenient since this allows us to calculate the derivative of the output probability with respect to an observation vector element. This derivative can then be extended to the lifter parameters by calculating the derivative of the element of the observation vector with respect to the lifter parameter. The lifter parameters can be trained, using these calculated values for the gradient descent optimization. After adaptation, the HMM is then retrained based on the dynamic cepstrum data obtained with the new lifter array. This training process is shown in Figure 2.

5.1. The derivatives of the loss function

The derivatives of the loss function can be calculated with respect to the lifter parameters as,

$$\frac{\partial L(\mathcal{O},\zeta)}{\partial \psi_h} = \sum_{r=1}^{R} \frac{\mathrm{d}\Gamma(d(\mathcal{O}_r,\zeta))}{\mathrm{d}d(\mathcal{O}_r,\zeta)} \times \frac{1}{TK} \sum_{t=1}^{T} \sum_{k=1}^{K} \left[-\frac{\partial g(\mathcal{O}_r,\zeta_c)}{\partial o_{r,t,k}} + \frac{\partial g(\mathcal{O}_r,\zeta_p)}{\partial o_{r,t,k}} \right] \frac{\partial o_{r,t,k}}{\partial \psi_h}$$

$$p = \underset{\mathrm{q,q} \neq \mathrm{c}}{\operatorname{argmax}}\ g(\mathcal{O}_r,\zeta_q) \tag{12}$$

where ψ_h denotes a lifter parameter of the lifter for $t-h$. K is the parameter dimension.

The derivatives of the discriminant function with respect to the observation are calculated as,

$$\frac{\partial g(\mathcal{O}_r,\zeta_p)}{\partial o_{r,t,k}} = \frac{1}{P_{s_t}(\mathbf{o}_{r,t})} \frac{\partial P_{s_t}(\mathbf{o}_{r,t})}{\partial o_{r,t,k}} \tag{13}$$

$$\frac{\partial P_{s_t}(\mathbf{o}_{r,t})}{\partial o_{r,t,k}} = \sum_{m=1}^{M} \lambda_{s_t,m} \times \prod_{k=1}^{K} \mathcal{M}(\mu_{s_t,m,k}, \varsigma^2_{s_t,m,k}, o_{r,t,k}) \left[\frac{-(o_{r,t,k} - \mu_{s_t,m,k})}{\varsigma^2_{s_t,m,k}} \right] \tag{14}$$

where $\mathcal{M}$ is a Gaussian probability density function of the mean μ and the standard deviation ς. λ is the mixture weight.

The derivatives of a component of an observation vector with respect to the lifter parameters are given by the derivatives of Eq. (1) with respect to the lifter parameters as,

$$\frac{\partial o_{r,t,k}}{\partial G(n)} = -c_k(t-n)\exp\left(-\frac{k^2}{2\sigma^2(n)}\right) \tag{15}$$

$$\frac{\partial o_{r,t,k}}{\partial \sigma(n)} = -c_k(t-n)\frac{G(n)k^2}{\sigma^3(n)}\exp\left(-\frac{k^2}{2\sigma^2(n)}\right) \tag{16}$$

After calculation of the derivatives of the loss function with respect to the lifter parameters, the lifter parameters can be adapted according to the Eq. (3).

6. SAMPLE SELECTION

An alternative to using the loss function, defined in the previous section, is to select which samples will be used to train the lifter parameters. The samples near the decision boundaries should be the most important in the training procedure. This paper proposes this new criterion of sample selection, instead of weighting the calculated derivatives for each sample according to their proximity to the decision boundary. The proximity of a correctly classified sample to a decision boundary can be expressed by the following measure:

$$V(\mathcal{O}_r) = \frac{P(\mathcal{O}_r \mid \zeta_p)}{P(\mathcal{O}_r \mid \zeta_c)} \tag{17}$$

This measure will approach one as the distance between a sample and the decision boundary approaches zero.

The influence of the misclassified and the correctly classified samples are balanced on the adaptation made to the lifter parameters by using an equal number of samples of both categories. The derivatives of all misclassified samples of a certain class were used. The same number of correctly classified samples with the smallest distances to the decision boundaries of the incorrectly classified classes, were selected. For example, if a /g/ sample was incorrectly classified as /b/, the erroneous sample was used as well as a correctly classified /g/ sample that was closest to the /g/-/b/ decision boundary. The lifter parameters were adapted according to:

$$\Delta\psi = \eta \left[\frac{\partial P(\mathcal{O} \mid \zeta_c)}{\partial \psi} - \frac{\partial P(\mathcal{O} \mid \zeta_p)}{\partial \psi} \right] \tag{18}$$

where ψ is a lifter parameter, $P(\mathcal{O} \mid \zeta_c)$ is the output probability of the model of the class of the sample and $P(\mathcal{O} \mid \zeta_p)$ is the output probability of the most competitive model of the sample.

7. EXPERIMENTS

The optimization of the lifter parameters was performed using the 2620 even-numbered words in the ATR Japanese speech database. The speech data for tests were collected from the odd-numbered 2620 Japanese words and the 115 sentences in the same database. The sampling frequency is 12 kHz. The speech data are converted to a 16th-order cepstrum sequence through 16th-order LPC analysis. The recognition results are all obtained using diagonal mixture Gaussian HMM's. Experiments were done on a six-phoneme recognition task for the consonants /b,d,g,m,n,N/.

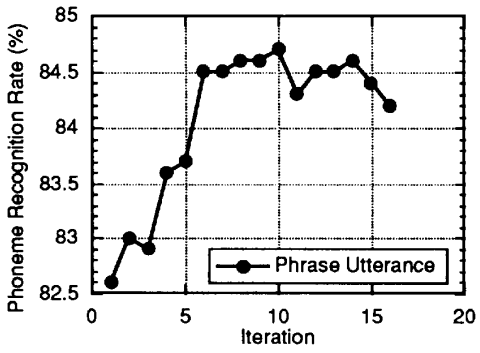

Figure 3. Recognition results by HMMs for speech data uttered in a different style, using the adapted lifter parameters, that were trained with a TDNN back end classifier.

	Initial Value		Optimized Value	
n	$G(n)$	$\sigma(n)$	$G(n)$	$\sigma(n)$
1	0.3	18	0.291755	18
2	0.21	17	0.220645	17
3	0.147	16	0.174029	16
4	0.1029	15	0.142169	15

Table 1. Optimized lifter parameters, starting with the heuristically selected parameters

The initial values for the dynamic cepstrum lifter array were set to the heuristically selected values reported in [1].

With a TDNN classifier, the recognizer was trained for 1000 iterations. The lifter parameters were not adapted during this period. After 1000 iterations, the lifter parameters were adapted, keeping the weights of the recognizer fixed. The results of the TDNN optimization were verified using an eight-mixture HMM. Figure 3 shows the results of the obtained lifter array applied to recognition of speech uttered in a different way.

When HMMs were used as the back end classifier, three mixture models were used for the adaptation of the lifter parameters. Figure 4 shows the recognition result of the obtained lifter array using speech uttered in a different style. These results show that a further increase in the recognition result can be obtained by the described method by starting with the heuristically selected parameters. The training from this point led to a local maximum. The training of the parameters after perturbation from this point is shown.

For the heuristically selected parameters, the gain of the lifters decays exponentially with increasing time interval between present and preceding spectra. After optimization, the exponential decay characteristic of the gain parameters of the lifters is dumped. The optimized set of parameters is shown in Table 1. The recognition rate was very insensitive to the standard deviation of the lifters.

The original cepstral representation of the speech data leads to a recognition result of 72.2% using a three-mixture HMM and 71.3% using an eight-mixture HMM.

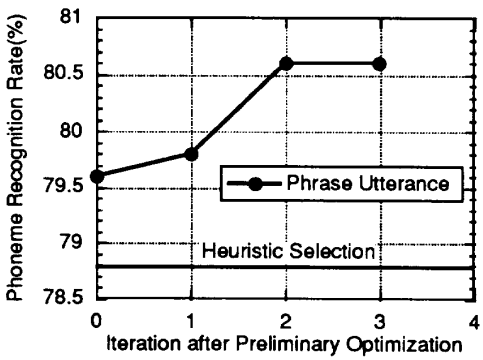

Figure 4. Recognition results for data, uttered in a different style, adapting the lifter parameters with an HMM back end classifier with a selected number of samples.

8. CONCLUSIONS

An algorithm for optimizing a set of lifters is proposed. The lifter array for the dynamic cepstrum is optimized by this algorithm. Two implementations are given, using a TDNN and a set of HMMs as back-end classifier. Both implementations clearly show improved recognition results not only for the training speech but also for the speech database in a different utterance style. The described optimization method can be used to train any type of parameterized lifter arrays, although this paper describes the optimization of a set of Gaussian-shaped lifters only.

ACKNOWLEDGMENT

The authors would like to thank Dr. Yoh'ichi Tohkura and Prof. A.J.M. Houtsma for providing this research opportunity. We would like to thank Dr. David Rainton for valuable discussions.

REFERENCES

[1] K. Aikawa, H. Singer, H. Kawahara, Y. Tohkura, "A dynamic cepstrum incorporating time-frequency masking and its application to continuous speech recognition," *Proc. ICASSP-93*, Vol. 2, pp. 668-671, 1993

[2] A. Biem, S. Katagiri, "Feature extraction based on minimum classification error/generalized probabilistic descent method," *Proc. ICASSP-93*, Vol. 2, pp. 275-278, 1993

[3] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, K. Lang, "Phoneme recognition using time-delay neural networks," *Proc. IEEE Trans. on Acoust., Speech, Signal Processing*, Vol. 37, No. 3, pp. 271-282, March 1989

[4] R. Lippmann, "An introduction to computing with neural nets," *Proc. IEEE Acoust., Speech and Signal Processing Magazine*, Vol. 4, No. 2, pp. 4-22, April 1987

[5] D. Rainton, S. Sagayama, "Minimum error classification training of HMMs," *J. Acoust. Soc. Jpn*, Vol. 13, No. 6, pp. 379-387, June 1992