

AN IMPROVEMENT FOR TRAINING EFFICIENCY OF SEMI-TIED COVARIANCE

Chen Si-Bao¹, Hu Yu¹, Luo Bin², Wang Ren-Hua¹

¹Iflytek Speech Lab, Department of Electronic Engineering and Information Science,
University of Science and Technology of China, Hefei

²School of Computer Science and Technology, Anhui University, Hefei

ABSTRACT

Semi-Tied Covariance (STC) is applied widely in speech recognition due to its feature de-correlation ability. Solving the transform matrices of STC is a nonlinear optimization problem. Gales proposed an efficient method by iteratively updating a row of transform matrices. However, it needs to solve cofactors of elements of a matrix row in two layers of loops. Directly solving them is very time-consuming. Based on the property that only one row is updated in each iteration, it can be found from algebraic procedures, that the inverse and determinant of transform matrix in current iteration can be obtained by simple multiplications and additions of those in the previous iteration, and the cofactor vector of a row is equal to the corresponding column of multiplication between the inverse and determinant. This clearly improves the training efficiency of STC. Experiments on the RM database show that the proposed iteration method achieves a 33.56% relative reduction of training time over original STC method.

Index Terms— Speech recognition, feature transformation, STC, Gales iteration, training efficiency

1. INTRODUCTION

In LVCSR, GMM-based CDHMMs have become prominent. However, too many model parameters, such as means and covariances, need to be estimated. Due to finity of training data, covariances usually take diagonal forms. It makes an assumption that features in different dimensions are irrelevant. But this assumption obviously disobeys the reality of features, such as MFCC and PLP.

Researchers have proposed many methods for solving the above problem. These methods can be categorized into two classes: one is restricting the structures of covariance matrices or inverse covariance (precision) matrices, to model the full covariance matrices; and the other is using feature transformation methods to eliminate the relevance of speech features between different dimensions.

In the majority, covariance/precision matrices in the first class methods are expressed as linear combinations of a small

set of prototype matrices that are shared across components. These prototypes and weights are estimated under ML sense. They need a great deal of training data and computational complexities are very high. In the second class, linear transformations, due to easy analysis and implementation, are adopted widely. That is, through some feature space rotations and scalings, different dimensions of speech features in new coordinates are made as irrelevant as possible. Major methods are: PCA-based KL transformation [1] and DCT [2]; maximizing class discriminative information based LDA [3], HDA [4] and PLDA [5]; ML-based MLLT [6] and HLDA [7], and so on. Moreover, transforming all feature vectors using only one transform matrix is not enough. Researchers proposed methods using more transform matrices to transform features, such as STC [8] and multiple HLDA [9].

Semi-tied covariance (STC), due to high recognition performance and stability, is widely adopted. Training STC to get transform matrices is a non-linear optimization problem, and its training efficiency is very important. Gales iteration greatly improves the training efficiency of STC. Moreover, Gales iteration method can be directly implemented into other training optimization problems, such as HLDA [7], MHLDA [9], CMLLR [10], and so its application value is huge. However, when training STC using Gales iteration, we need to compute cofactors of one row of matrix in two layers of loops. Computing cofactors directly is very time-consuming. In real applications, matrix decomposition is usually used to get the inverse and determinant of matrix, and then cofactors. Based on the fact that only one row is updated in each inner-loop of Gales iteration, the inverse and determinant of matrix in current iteration can be obtained by simple multiplications and additions of those in the previous iteration, and the cofactor vector of a row is equal to the corresponding column of multiplication between the inverse and determinant. Using algebraic induction procedures, we propose a method of fast computation of cofactors in two layers of loops of Gales iteration, which will improve the training efficiency of STC.

The paper is organized as follows: STC and Gales iteration are reviewed in Section 2; A general property of the inverse and determinant is given in Section 3 and then fast computation of cofactors in two layers of loops of Gales iteration is proposed; Section 4 details experiments on the RM

Thanks to the China National 863 Program (No. 2004AA114030) and National Natural Science Foundation of China (No.60772122) for funding.

database to show the improvements in training efficiency of STC; Conclusions are given in Section 5.

2. STC AND GALES ITERATION

2.1. Semi-tied covariance

Semi-tied covariance (STC) [8] assumes that all Gaussian components in HMM can be classified into several mutually exclusive classes, such as regression classes. All covariance matrices of Gaussian components take the following form:

$$\Sigma^{(m)} = \mathbf{H}^{(r)} \Sigma_{diag}^{(m)} \mathbf{H}^{(r)\top}, \quad (1)$$

where, m is the index of Gaussian component, r is the index of class, $\Sigma_{diag}^{(m)}$ is the special diagonal covariance matrix of Gaussian m , $\mathbf{H}^{(r)}$ is the shared square matrix of all covariances of Gaussian components in class r . Let $\mathbf{A}^{(r)} = \mathbf{H}^{(r)-1}$, then $\mathbf{A}^{(r)}$ is the transform matrix belonging to class r in the feature space.

When estimating $\mathbf{A}^{(r)}$ and $\Sigma_{diag}^{(m)}$ in ML sense, the auxiliary function is:

$$Q(\theta, \hat{\theta}) = \sum_{m \in M^{(r)}, \tau} \gamma_m(\tau) \{ \log |\hat{\mathbf{A}}^{(r)}|^2 - \log |\hat{\Sigma}_{diag}^{(m)}| - (o(\tau) - \hat{\mu}^{(m)})^\top \hat{\mathbf{A}}^{(r)\top} \hat{\Sigma}_{diag}^{(m)-1} \hat{\mathbf{A}}^{(r)} (o(\tau) - \hat{\mu}^{(m)}) \}, \quad (2)$$

where, $M^{(r)}$ is the set of Gaussian components belonging to class r , τ is the index of time, $o(\tau)$ is the n -dimensional observed feature vector at time τ , $\gamma_m(\tau)$ is the posterior of Gaussian m on feature $o(\tau)$ at time τ , $\hat{\mu}^{(m)} = \frac{\sum_{\tau} \gamma_m(\tau) o(\tau)}{\sum_{\tau} \gamma_m(\tau)}$ is the mean estimate of Gaussian m . When all the model parameters are optimized simultaneously, the auxiliary function can be rewritten as:

$$Q(\theta, \hat{\theta}) = \sum_{m \in M^{(r)}, \tau} \gamma_m(\tau) \{ \log |\hat{\mathbf{A}}^{(r)}|^2 - \log |diag(\hat{\mathbf{A}}^{(r)} \mathbf{W}^{(m)} \hat{\mathbf{A}}^{(r)\top})| \} - n\beta, \quad (3)$$

where,

$$\mathbf{W}^{(m)} = \frac{\sum_{\tau} \gamma_m(\tau) (o(\tau) - \hat{\mu}^{(m)}) (o(\tau) - \hat{\mu}^{(m)})^\top}{\sum_{\tau} \gamma_m(\tau)}$$

is the sample estimate of covariance of Gaussian m ,

$$\beta = \sum_{m \in M^{(r)}, \tau} \gamma_m(\tau)$$

is the occupation of class r .

The re-estimation of Gaussian weights, transition probabilities and Gaussian means in the model are the same as that of usual HMM. The ML estimation of Gaussian diagonal covariance is:

$$\hat{\Sigma}_{diag}^{(m)} = diag(\hat{\mathbf{A}}^{(r)} \mathbf{W}^{(m)} \hat{\mathbf{A}}^{(r)\top}). \quad (4)$$

However, the formula (3), to estimate transform matrix $\hat{\mathbf{A}}^{(r)}$, is a non-linear optimization problem. Some numerical optimization methods, such as steepest descend or quasi-Newton, can be adopted to solve it, but the computational complexity is very high.

2.2. Gales iteration

Gales [8] proposed the following iteration method to estimate diagonal covariance and transform matrix of STC:

- 1) Estimate Gaussian weights, transition probabilities and Gaussian means in the model, which are independent of the other model parameters;
- 2) Fix the current transform matrix estimate $\hat{\mathbf{A}}^{(r)}$, estimate the diagonal covariances $\{\hat{\Sigma}_{diag}^{(m)}, m \in M^{(r)}\}$ using formula (4);
- 3) Fix the diagonal covariances $\{\hat{\Sigma}_{diag}^{(m)}, m \in M^{(r)}\}$, update the transform matrix $\hat{\mathbf{A}}^{(r)}$ iteratively row by row using the following formula:

$$\hat{\mathbf{a}}_i^{(r)} = \mathbf{c}_i \mathbf{G}^{(ri)-1} \sqrt{\frac{\beta}{\mathbf{c}_i \mathbf{G}^{(ri)-1} \mathbf{c}_i^\top}}, \quad (5)$$

where,

$$\mathbf{G}^{(ri)} = \sum_{m \in M^{(r)}} \frac{\mathbf{W}^{(m)} \sum_{\tau} \gamma_m(\tau)}{\hat{\sigma}_{diag,i}^{(m)2}},$$

$\hat{\sigma}_{diag,i}^{(m)2}$ is the i 'th principal diagonal element of $\hat{\Sigma}_{diag}^{(m)}$, \mathbf{c}_i is a row vector consisting of cofactors of the i 'th row elements of $\hat{\mathbf{A}}^{(r)}$;

- 4) Return to step 2) until convergence.

In the above iteration, transform matrix $\hat{\mathbf{A}}^{(r)}$ of different classes can be estimated independently. In step 3), when updating a row of $\hat{\mathbf{A}}^{(r)}$, computing cofactors of current row elements needs to fix other rows of $\hat{\mathbf{A}}^{(r)}$.

Experiments show that Gales iteration is faster than other numerical optimization methods. Moreover, this kind of iteration can be applied to other non-linear optimization problems such as HLDA [7], MHLDA [9], CMLLR [10], and so on. Therefore, its application value is very huge.

Gales [8] doesn't mention how to compute cofactors of elements of a matrix row in two layers of loops. Computing cofactor directly has the computational complexity of $O(n!)$, it cannot be used in practical application. Usually, matrix decomposition is adopted to compute them. HTK-3.4 toolbox [11] takes LU decomposition of matrix, and its computational complexity is $O(n^3)$. However, when there are many classes, training STC transform matrices still needs much time.

3. ITERATIVELY COMPUTING THE INVERSE AND DETERMINANT OF A MATRIX

Before introducing our improved iteration method to solve STC transform matrices, we give a general property of the inverse and determinant of a matrix. For simplifying notations and avoiding confusion, we omit the superscript class index $^{(r)}$ and hat notation of estimation of STC transform matrix estimate $\hat{\mathbf{A}}^{(r)}$, and just denote \mathbf{A} in the rest of this paper.

3.1. A property of inverse and determinant of matrix

Assuming \mathbf{A} is an reversible square matrix of size $(n \times n)$, \mathbf{a} and \mathbf{b} are both column vectors of size $(n \times 1)$, then we have

$$(\mathbf{A} + \mathbf{a}\mathbf{b}^\top)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{a}\mathbf{b}^\top\mathbf{A}^{-1}}{1 + \mathbf{b}^\top\mathbf{A}^{-1}\mathbf{a}}, \quad (6)$$

$$|\mathbf{A} + \mathbf{a}\mathbf{b}^\top| = |\mathbf{A}|(1 + \mathbf{b}^\top\mathbf{A}^{-1}\mathbf{a}). \quad (7)$$

This property is a special case of Sherman-Morrison-Woodbury formula [12], and the formula (6) can be easily proved by verifying the equation

$$(\mathbf{A} + \mathbf{a}\mathbf{b}^\top)(\mathbf{A} + \mathbf{a}\mathbf{b}^\top)^{-1} = \mathbf{I}$$

being correct, where \mathbf{I} is identity matrix of order n . Bring formula (6) into the left side, multiply into brackets and the identity matrix will be obtained. The proof of (7) needs to construct a $(n+1) \times (n+1)$ size matrix $\begin{pmatrix} 1 & -\mathbf{b}^\top \\ \mathbf{a} & \mathbf{A} \end{pmatrix}$, then its determinant

$$\begin{aligned} \begin{vmatrix} 1 & -\mathbf{b}^\top \\ \mathbf{a} & \mathbf{A} \end{vmatrix} &= \begin{vmatrix} 1 & \mathbf{b}^\top\mathbf{A}^{-1} \\ \mathbf{0} & \mathbf{I} \end{vmatrix} \begin{vmatrix} 1 & -\mathbf{b}^\top \\ \mathbf{a} & \mathbf{A} \end{vmatrix} \\ &= \begin{vmatrix} 1 + \mathbf{b}^\top\mathbf{A}^{-1}\mathbf{a} & \mathbf{0}^\top \\ \mathbf{a} & \mathbf{A} \end{vmatrix} = (1 + \mathbf{b}^\top\mathbf{A}^{-1}\mathbf{a})|\mathbf{A}|, \end{aligned}$$

where $\mathbf{0}$ is a column vector of size $(n \times 1)$ with all its elements being zero. On the other side,

$$\begin{aligned} \begin{vmatrix} 1 & -\mathbf{b}^\top \\ \mathbf{a} & \mathbf{A} \end{vmatrix} &= \begin{vmatrix} 1 & -\mathbf{b}^\top \\ \mathbf{a} & \mathbf{A} \end{vmatrix} \begin{vmatrix} 1 & \mathbf{b}^\top \\ \mathbf{0} & \mathbf{I} \end{vmatrix} \\ &= \begin{vmatrix} 1 & \mathbf{0}^\top \\ \mathbf{a} & \mathbf{A} + \mathbf{a}\mathbf{b}^\top \end{vmatrix} = |\mathbf{A} + \mathbf{a}\mathbf{b}^\top|. \end{aligned}$$

Therefore, the second formula is also correct. The proof of the property (6) and (7) of the inverse and determinant of a matrix is now complete.

3.2. Iteratively computing the inverse and determinant of an STC transform matrix

With the above property of inverse and determinant of matrix, we can compute the inverse and determinant of an STC transform matrix in current iteration using those in the previous iteration.

Let $\mathbf{a} = \mathbf{e}_i$, $\mathbf{b}^\top = (\mathbf{a}_i^{(k+1)} - \mathbf{a}_i^{(k)})$, where, \mathbf{e}_i is a column vector of size $(n \times 1)$ with the i 'th element being one, the others being zero, $\mathbf{a}_i^{(k)}$ is the i 'th $(1 \times n)$ row of transform matrix estimate $\mathbf{A}^{(k)}$ in the k 'th iteration. Then we have

$$\mathbf{A}^{(k)} + \mathbf{a}\mathbf{b}^\top = \mathbf{A}^{(k)} + \mathbf{e}_i(\mathbf{a}_i^{(k+1)} - \mathbf{a}_i^{(k)}) = \mathbf{A}^{(k+1)}. \quad (8)$$

Therefore, we only need to compute the inverse and determinant of the transform matrix in the initial iteration. In the following iterations, updated inverse and determinants of transform matrix can be computed iteration by iteration using

formulae (6) and (7). The computational complexity of solving the inverse and determinant using matrix decomposition is $O(n^3)$, while that of using formulae (6) and (7) is $O(n^2)$.

Once we have the inverse \mathbf{A}^{-1} and the determinant $|\mathbf{A}|$ of the transform matrix \mathbf{A} , then the vector \mathbf{c}_i , consisting of cofactors of elements of the i 'th row \mathbf{a}_i of \mathbf{A} , is equal to the i 'th row of multiplication of the determinant and transposed inverse $|\mathbf{A}|\mathbf{A}^{-\top}$. That is

$$[\mathbf{c}_1^\top, \mathbf{c}_2^\top, \dots, \mathbf{c}_n^\top] = |\mathbf{A}|\mathbf{A}^{-1}. \quad (9)$$

The outer-loop of Gales iteration for training STC matrices is the alternate iteration between step 2) and 3), which is the alternate update between diagonal covariance components $\{\Sigma_{diag}^{(m)}\}$ and transform matrix \mathbf{A} . The inner-loop of Gales iteration is in step 3), which updates \mathbf{A} iteratively row by row. Using formulae (6) and (7), the inner-loop can be summarized as the follows:

- 3.1) Set $k = 0$, compute the inverse and determinant of initial transform matrix $\mathbf{A}^{(k)}$ using LU decomposition;
- 3.2) Let $i = (k+1)\%n$, Compute $\mathbf{c}_i^{(k+1)}$ using formula (9);
- 3.3) Compute $\mathbf{a}_i^{(k+1)}$ using formula (5);
- 3.4) Compute $\mathbf{A}^{(k+1)}$ using formula (8);
- 3.5) Compute the inverse and determinant of $\mathbf{A}^{(k+1)}$ using formulae (6) and (7);
- 3.6) Set $k = k + 1$ and go to step 3.2) until convergence.

Computing the inverses and determinants of transform matrices using formulae (6) and (7) in two layers of loops of STC training iterations obviously can improve the training efficiency of STC.

4. EXPERIMENTS

To evaluate the improved STC iteration, we examined the training time and recognition performance of STC between the improved iteration, which is based on iteratively updating the inverse and determinant (iterInvDet), and the usual iteration, which is based on LU matrix decomposition (HTK-3.4), with different regression classes on the RM database [13]. We chose a 39-dimensional vector, including 12 MFCC, log energy, and their first- and second-order differences, as training features. All the training procedures are based on Cambridge's HTK-3.4 toolbox [11].

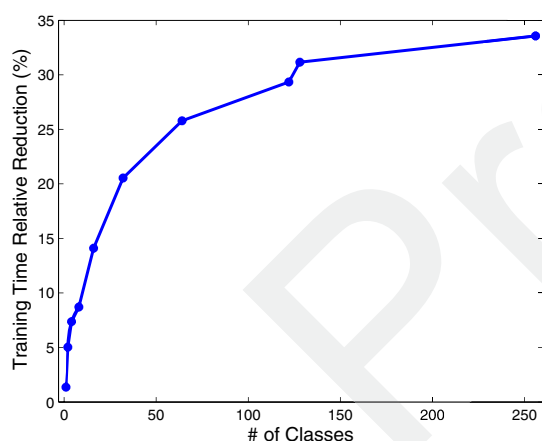
The details of the RM database can be found in [13]. We used the standard NIST/RM SI-109 training set to train a tied-state tri-phone model. Each tied-state has six Gaussian components. A total set of all test sets feb89, oct89, feb91, sep92 is used for evaluation.

In experiments, when choosing the same regression classes, we found that the improved iteration method and HTK iteration generate the same STC transform matrices and models. Therefore, the recognition performances should be the same, and so were tested only once. Figure 1 shows the

Table 1. Comparison of STC Training Efficiency on RM Database

| # of classes | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 122 | 128 | 256 |
|-----------------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Acc (%) | 96.11 | 96.04 | 96.40 | 96.27 | 96.43 | 96.40 | 96.37 | 96.16 | 96.09 | 96.01 |
| Time (HTK-3.4) | 02:27 | 02:39 | 02:43 | 03:04 | 03:54 | 05:36 | 08:59 | 15:10 | 16:54 | 32:20 |
| Time (iterInvDet) | 02:25 | 02:31 | 02:31 | 02:48 | 03:21 | 04:27 | 06:40 | 10:43 | 11:38 | 21:29 |
| Time Reduction | 00:02 | 00:08 | 00:12 | 00:16 | 00:33 | 01:09 | 02:19 | 04:27 | 05:16 | 10:51 |
| Time Relative Reduction (%) | 1.36 | 5.03 | 7.36 | 8.70 | 14.10 | 20.54 | 25.79 | 29.34 | 31.16 | 33.56 |

variation of training time relative reduction between the proposed iteration method and HTK-3.4 iteration along different number of classes, and Table 1 gives the detailed recognition performance, training time and relative reduction of training time. From the table, we can see that training efficiency increases more as the number of classes becomes larger. This is because there are other procedures, such as file I/O of features and models, and updating of diagonal covariances in the STC training procedure. Increasing the number of classes doesn't need more time in these areas, while needs to estimate more transform matrices. Therefore, the more classes, the more increase of training efficiency. From the table, we can see that the improved STC iteration takes one third off the original time, showing the validity of the proposed STC iteration.

**Fig. 1.** Training Time Relative Reduction of STC between the proposed method and that of LU decomposition.

5. CONCLUSIONS

Solving STC transform matrices is a non-linear optimization problem. Gales iteration, updating matrix row by row iteratively, needs to compute cofactors of a row elements in two layers of loops. Directly computing them is impractical and computing using matrix decomposition is still too time consuming. Based on the fact that only one row is updated in an iteration and through algebraic procedures, cofactors were computed from the inverse and determinant of the matrix, which were computed from those in the previous iteration. This iteration method greatly improves the training efficiency of STC. Experiments on the RM database show the validity

of the proposed method.

6. REFERENCES

- [1] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic, New York, 1972.
- [2] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 28, no. 4, pp. 357–366, 1980.
- [3] R. Haeb-Umbach and H. Ney, "Linear discriminant analysis for improved large vocabulary continuous speech recognition," in *ICASSP*. IEEE, 1992, vol. I, pp. 13–16.
- [4] G. Saon, M. Padmanabhan, R. Gopinath, and S. Chen, "Maximum likelihood discriminant feature spaces," in *ICASSP*. IEEE, 2000, vol. II, pp. 1129–1132.
- [5] M. Sakai, N. Kitaoka, and S. Nakagawa, "Generalization of linear discriminant analysis used in segmental unit input HMM for speech recognition," in *ICASSP*. IEEE, 2007, vol. IV, pp. 333–337.
- [6] R. A. Gopinath, "Maximum likelihood modeling with gaussian distributions for classification," in *ICASSP*. IEEE, Seattle, WA, May 1998, vol. 2, pp. 661–664.
- [7] N. Kumar, *Investigation of silicon-auditory models and generalization of linear discriminant analysis for improved speech recognition*, Ph.D. thesis, Johns Hopkins Univ., Baltimore, MD, 1997.
- [8] M. J. F. Gales, "Semi-tied covariance matrices for hidden markov models," *IEEE Trans. Speech Audio Process.*, vol. 7, no. 3, pp. 272–281, 1999.
- [9] M. J. F. Gales, "Maximum likelihood multiple subspace projections for hidden markov models," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 2, pp. 37–47, 2002.
- [10] M. J. F. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1998.
- [11] S. Young, G. Evermann, M. Gales, and et al., *The HTK Book (for HTK Version 3.4)*, Available: <http://htk.eng.cam.ac.uk/>, 2006.
- [12] G. H. Golub and C. F. Van Loan, *Matrix computations (3rd ed.)*, Johns Hopkins University Press, 1996.
- [13] P. Price, W. M. Fisher, J. Bernstein, and et al., "The DARPA 1000-word resource management database for continuous speech recognition," in *ICASSP*. IEEE, 1988, vol. I, pp. 651–654.