A COMPACT SEMIDEFINITE PROGRAMMING (SDP) FORMULATION FOR LARGE MARGIN ESTIMATION OF HMMS IN SPEECH RECOGNITION

Yan Yin, Hui Jiang

Department of Computer Science and Engineering, York University
4700 Keele Street, Toronto, Ontario M3J 1P3, CANADA
{yyin,hj}@cse.yorku.ca

ABSTRACT

In this paper, we study a new semidefinite programming (SDP) formulation to improve optimization efficiency for large margin estimation (LME) of HMMs in speech recognition. We re-formulate the same LME problem as smaller-scale SDP problems to speed up the SDP-based LME training, especially for large model sets. In the new formulation, instead of building the SDP problem from a single huge variable matrix, we consider to formulate the SDP problem based on many small independent variable matrices, each of which is built separately from a Gaussian mean vector. Moreover, we propose to further decompose feature vectors and Gaussian mean vectors according to static, delta and accelerate components to build even more compact variable matrices. This method can significantly reduce the total number of free variables and result in much smaller SDP problem even for the same model set. The proposed new LME/SDP methods have been evaluated on a connected digit string recognition task using the TIDIGITS database. Experimental results show that it can significantly improve optimization efficiency (about 30-50 times faster for large model sets) and meanwhile it can provide slightly better optimization accuracy and recognition performance than our previous SDP formulation.

Index Terms— Automatic Speech Recognition, Discriminative training, Large Margin Estimation (LME), Convex Optimization, Semidefinite Programming (SDP), Convex Relaxation

1. INTRODUCTION

In the past few years, it has been shown that discriminative training techniques, such as maximum mutual information (MMI) and minimum classification error (MCE), can significantly improve speech recognition performance over the conventional maximum likelihood (ML) estimation. Recently, we have proposed a magin-based discriminative training method, namely large margin estimation (LME), for speech recognition [8, 4, 11], where Gaussian mixture continuous density hidden Markov models (CDHMM) are estimated based on the principle of maximizing the minimum margin. From the theoretical results in machine learning, a large margin classifier implies good generalization power and generally yields much lower generalization errors in unseen test data. As shown in [8], estimation of large margin HMMs turns out to be a constrained minimax optimization problem. After we approximate the problem with differentiable functions, the constrained minimax optimization problem can be solved by gradient descent methods, such as *iterative localized optimization (ILO)* in [8], *constrained joint optimization (CJO)* method in [4]. However, the gradient descent method can only lead to locally optimal solution which highly depends on the initial models and it is also hard to run in practice because it is necessary to manually tune a number of sensitive parameters in experiments.

More recently, we have proposed to solve the LME problem using convex optimization. In convex optimization problems, any local optimum is also globally optimal. Because of this, it is relatively simple and efficient to solve very large scale convex optimization problems with guaranteed convergence to good solutions. In [11], we have formulated the LME problem of Gaussian mixture HMMs as a semidefinite programming (SDP) under some relaxation conditions. Then the derived SDP problem can be solved by many SDP algorithms, such as [1] and many others, which lead to the globally optimal solution since SDP is a well-defined convex optimization. It has been demonstrated that the SDP method outperforms all other methods and it has achieved one of the best performances on the TIDIGITS task [11]. However, optimization time of the LME/SDP method increases dramatically as model size grows because size of the SDP variable matrix in [11] (i.e., Z) roughly equals the square of total number of Gaussians in the model set. In [11], we have successfully managed to handle an HMM set consisting of about 4k Gaussians but the LME/SDP method in [11] is unlikely to extend directly to other larger scale speech recognition tasks which involve tens or even hundreds of thousands of Gaussians. To apply the LME method to larger scale speech recognition tasks, in [13], we have proposed to formulate LME as another simpler convex optimization problem, i.e. second order cone programming (SOCP), under some different relaxation conditions. It has been shown that the LME/SOCP formulation in [13] significantly improves optimization efficiency and it yields a much faster LME training method, especially for large model sets. As demonstrated in many engineering problems, the SDP relaxation is normally tighter than its SOCP counterpart. As the result, the SDP method yields better modeling and optimization accuracy than an SOCP method. For example, the LME/SOCP gives slightly lower recognition accuracy than the LME/SDP method as shown in [13] due to the looser relaxation condition in SOCP.

In this paper, we study a different way to improve optimization efficiency for the LME training of Gaussian mixture HMMs in speech recognition. As opposed to formulating LME as another simpler convex optimization problem, we still rely on the SDP framework due to its inherent high accuracy but we consider to formulate the same LME problem into a smaller SDP problem than the one in [11] in number of free variables and constraints. More specifically, in this work, we propose to re-formulate LME as a different SDP problem based on many small variable matrices instead of using only a single large matrix as in [11]. In the new formulation, each small matrix is constructed based on one Gaussian mean vector in the model set and the resultant SDP is performed with respect to all these small matrices subject to the constraint that all of these

small matrices are symmetric and positive semi-definite. In this way, we can significantly reduce the total number of free variables in the SDP problem, especially for large model sets. Moreover, when we construct the variable matrix for each Gaussian, we can also further decompose feature vectors and Gaussian mean vectors according to static, delta, acceleration components to build an even more compact matrix for each Gaussian component. The newly proposed compact LME/SDP formulation has been examined in a connected digit string recognition task on the TIDIGITS database. Experimental results show that the LME/SDP method can significantly improve optimization efficiency (about 30-50 times faster for large model sets) and also yield slightly better optimization accuracy and recognition performance than our previous SDP formulation.

2. LARGE MARGIN ESTIMATION (LME) OF HMMS

As in [8, 4], separation margin for a speech utterance $\mathbf{X}_i$ in speech recognition is defined as:

$$\begin{aligned} d(\mathbf{X}_i) &= \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) - \max_{j\in\Omega\ j\neq W_i} \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_j) \\ &= \min_{j\in\Omega\ j\neq W_i} \Big[\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) - \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_j)\Big] \end{aligned} \quad (1)$$

where Ω denotes the set of all possible words, $\boldsymbol{\lambda}_W$ denotes HMM representing a word or word sequence W, W_i is the true word identity for $\mathbf{X}_i$ and $\mathcal{F}(\mathbf{X}|\boldsymbol{\lambda}_W)$ is called the discriminant function, which is usually calculated in the logarithm scale: $\mathcal{F}(\mathbf{X}|\boldsymbol{\lambda}_W) = \log[p(W)\cdot p(\mathbf{X}|\boldsymbol{\lambda}_W)]$. In this work, we are only interested in estimating HMM $\boldsymbol{\lambda}_W$ and assume $p(W)$ is fixed.

Given a set of training data $\mathcal{D} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_R\}$, we usually know the true word identities for all utterances in $\mathcal{D}$, denoted as $\mathcal{L} = \{W_1, W_2, \cdots, W_R\}$. The *support vector set* $\mathcal{S}$ is defined as:

$$\mathcal{S} = \{\mathbf{X}_i \mid \mathbf{X}_i \in \mathcal{D} \text{ and } 0 \leq d(\mathbf{X}_i) \leq \gamma\} \quad (2)$$

where $\gamma > 0$ is a pre-set positive number. All utterances in $\mathcal{S}$ are relatively close to the classification boundary even though all of them locate in the right decision regions.

The large margin principle leads to estimating all HMM models, denoted as Λ, based on the criterion of maximizing the minimum margin of all support tokens, which is named as large margin estimation (LME) of HMM.

$$\begin{aligned} \Lambda^* &= \arg\max_{\Lambda} \min_{\mathbf{X}_i\in\mathcal{S}} d(\mathbf{X}_i) \\ &= \arg\min_{\Lambda} \max_{\mathbf{X}_i\in\mathcal{S}} \max_{j\in\Omega\ j\neq W_i} \Big[\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_j) - \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i})\Big] \end{aligned} \quad (3)$$

Assume that the entire model set Λ consists of a total of $\mathcal{K}$ Gaussian mixtures, each of which is denoted as $\mathcal{N}(\boldsymbol{\mu}_k, \Sigma_k)$ with $k \in (1, 2, \ldots, \mathcal{K})$. For simplicity, we only consider diagonal covariance matrix, i.e., $\Sigma_k = \text{diag}(\sigma_{k1}^2, \cdots, \sigma_{kD}^2)$ where D stands for feature dimension. Given any speech utterance $\mathbf{X}_i = \{\mathbf{x}_{i1}, \mathbf{x}_{i2}, \cdots, \mathbf{x}_{iT}\}$, if we only estimate Gaussian mean vectors, discriminant function $\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda})$ in eq.(1) can be approximated based on the Viterbi path as:

$$\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) \approx c_i - \frac{1}{2}\sum_{t=1}^{T}\sum_{d=1}^{D}\Big[\frac{(x_{itd} - \mu_{i_t d})^2}{\sigma_{i_t d}^2}\Big] \quad (4)$$

where the optimal Viterbi path of $\mathbf{X}_i$ against $\boldsymbol{\lambda}_{W_i}$ is denoted as a sequence of Gaussians along the path, i.e., $\mathbf{i} = \{i_1, i_2, \cdots, i_T\}$ with $i_t \in (1, 2, \ldots, \mathcal{K})$, and c_i is a constant independent of all Gaussian means.

Since the margin as defined in eq.(1) is actually unbounded for Gaussian mixture HMMs, as in [11] we adopt the following spherical constraint to guarantee the boundedness of margin:

$$R(\Lambda) = \sum_{k=1}^{\mathcal{K}}\sum_{d=1}^{D}\frac{(\mu_{kd} - \mu_{kd}^{(0)})^2}{\sigma_{kd}^2} \leq r^2 \quad (5)$$

where r is a pre-set constant, and $\mu_{kd}^{(0)}$ is also a constant which is set to be the value of μ_{kd} in the initial models. As shown in [11], the minimax optimization problem in eq.(3) becomes solvable under the constraint eq.(5). Following [11], if we introduce a new variable $-\rho$ ($\rho > 0$) as the common upper bound for all terms in the minimax optimization, we can convert the minimax optimization in eq.(3) into an equivalent minimization problem as follows:

Problem 1

$$\Lambda^* = \arg\min_{\Lambda,\rho} -\rho \quad (6)$$

subject to

$$\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_j) - \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) \leq -\rho \quad (7)$$

$$\text{for all } \mathbf{X}_i \in \mathcal{S} \text{ and } j \in \Omega \text{ and } j \neq W_i\,,$$

$$R(\Lambda) = \sum_{k=1}^{\mathcal{K}}\sum_{d=1}^{D}\frac{(\mu_{kd} - \mu_{kd}^{(0)})^2}{\sigma_{kd}^2} \leq r^2, \quad (8)$$

$$\rho \geq 0. \quad (9)$$

3. PREVIOUS SDP FORMULATION FOR LME

In [10, 11, 12], we have formulated the above LME Problem 1 as an SDP problem. In this section, we first briefly review the LME/SDP formulation previously proposed in [10, 11, 12].

As we know, semidefinite programming (SDP) is a class of convex optimization problems and SDP can be viewed as a generalization of linear programming. In an SDP problem, we minimize a linear function of symmetric matrices in a positive semidefinite matrix cone subject to some affine constraints:

$$\text{Minimize} \qquad \sum_{j=1}^{J} C_j \cdot X_j \quad (10)$$

subject to

$$\sum_{j=1}^{J} A_{ij} \cdot X_j \leq b_i, \qquad i = 1, \ldots, I, \qquad X_j \succeq 0. \quad (11)$$

where $X_j \succeq 0$ means each variable X_j is a symmetric positive semidefinite matrix. A_{ij} and C_j are real symmetric matrices with the same dimension as X_j, b_i is a scalar constant, and $X \cdot Y$ denotes the inner product of two symmetric matrices as: $X \cdot Y = \sum_{i,j} x_{ij}y_{ij}$.

Here, we introduce some notation: $\mathbf{e}_i$ is a $\mathcal{K}$-dimensional vector with -1 at the i-th position, and zero everywhere else. A column vector $\mathbf{x}$ is written as $\mathbf{x} = (x_1; x_2; \ldots; x_n)$ and a row vector as $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. I_D is a $D \times D$ identity matrix.

For notational convenience, we denote each normalized Gaussian mean vector (in column), $\tilde{\boldsymbol{\mu}}_k$ for all $k \in (1, \cdots, \mathcal{K})$ as:

$$\tilde{\boldsymbol{\mu}}_k := \Big(\frac{\mu_{k1}}{\sigma_{k1}}; \frac{\mu_{k2}}{\sigma_{k2}}; \ldots; \frac{\mu_{kD}}{\sigma_{kD}}\Big). \quad (12)$$

Then, we construct a matrix U by concatenating all normalized Gaussian mean vectors as columns:

$$U := (\tilde{\boldsymbol{\mu}}_1, \tilde{\boldsymbol{\mu}}_2, \ldots, \tilde{\boldsymbol{\mu}}_{\mathcal{K}}). \tag{13}$$

Next, we represent discriminant function $\mathcal{F}(\mathbf{X}|\boldsymbol{\lambda})$ in eq.(4) in matrix form using U:

$$\begin{aligned}\mathcal{F}(\mathbf{X}|\boldsymbol{\lambda}) &= c - \frac{1}{2}\sum_{t=1}^{T}(\tilde{\mathbf{x}}_t - \tilde{\boldsymbol{\mu}}_{i_t})'(\tilde{\mathbf{x}}_t - \tilde{\boldsymbol{\mu}}_{i_t}) \\ &= c - \frac{1}{2}\sum_{t=1}^{T}(\tilde{\mathbf{x}}_t; \mathbf{e}_{i_t})'(I_D, U)'(I_D, U)(\tilde{\mathbf{x}}_t; \mathbf{e}_{i_t}) \\ &= -A \cdot Z + c \end{aligned} \tag{14}$$

where the Viterbi path is denoted as $\mathbf{i} = \{i_1, \cdots, i_T\}$ and $\tilde{\mathbf{x}}_t$ denotes a column feature vector normalized with Gaussian variance along the path as:

$$\tilde{\mathbf{x}}_t := \left(\frac{x_{t1}}{\sigma_{i_t 1}};\ \frac{x_{t2}}{\sigma_{i_t 2}};\ \ldots;\ \frac{x_{tD}}{\sigma_{i_t D}}\right) \tag{15}$$

and

$$A = \frac{1}{2}\sum_{t=1}^{T}(\tilde{\mathbf{x}}_t; \mathbf{e}_{i_t})(\tilde{\mathbf{x}}_t; \mathbf{e}_{i_t})' \tag{16}$$

$$Z = \begin{pmatrix} I_D & U \\ U' & Y \end{pmatrix} \qquad Y = U'U. \tag{17}$$

It is straightforward to convert the constraint in eq. (7) into the following form:

$$\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_j) - \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) = A_{ij} \cdot Z - c_{ij} \leq -\rho \tag{18}$$

where $A_{ij} = A_i - A_j$ with A_i and A_j calculated according to eq.(16) based on the Viterbi decoding paths $\mathbf{i}$ in $\boldsymbol{\lambda}_{W_i}$ and $\mathbf{j}$ in $\boldsymbol{\lambda}_j$, respectively.

Similarly, $R(\Lambda)$ in eq.(8) can also be re-written as the following matrix form using Z:

$$\begin{aligned} R(\Lambda) &= \sum_{k=1}^{L}(\tilde{\boldsymbol{\mu}}_k - \tilde{\boldsymbol{\mu}}_k^{(0)})'(\tilde{\boldsymbol{\mu}}_k - \tilde{\boldsymbol{\mu}}_k^{(0)}) \\ &= Q \cdot Z \leq r^2 \end{aligned} \tag{19}$$

where $Q = \sum_{k=1}^{\mathcal{K}}(\tilde{\boldsymbol{\mu}}_k^{(0)}; \mathbf{e}_k)(\tilde{\boldsymbol{\mu}}_k^{(0)}; \mathbf{e}_k)'$, and $\tilde{\boldsymbol{\mu}}_{kd}^{(0)}$ is calculated in eq. (12) by using initial model parameters.

To convert the LME problem into an SDP, we must ensure all constraints are convex. However, the constraint $Y = U'U$ in eq.(17) is not convex. Following [3], if we relax the constraint $Y = U'U$ to $Y - U'U \succeq 0$, we are able to make Z a positive semidefinite matrix since $Y - U'U \succeq 0$ and $Z = \begin{pmatrix} I_D & U \\ U' & Y \end{pmatrix} \succeq 0$ are equivalent.

Finally, we can convert the original LME problem as the following SDP program:

Problem 2

$$\Lambda^* = \arg\min_{\Lambda,\rho} \quad -\rho \tag{20}$$

subject to

$$A_{ij} \cdot Z + \rho \leq c_{ij}, \tag{21}$$

$$\text{for all } \mathbf{X}_i \in \mathcal{S} \text{ and } j \in \Omega \text{ but } j \neq W_i$$

$$Q \cdot Z \leq r^2, \tag{22}$$

$$Z \succeq 0 \qquad \rho \geq 0, \tag{23}$$

$$Z_{1:D,1:D} = I_D. \tag{24}$$

The constraint in eq.(24) is imposed to ensure that the top-left corner of Z is an identity matrix as required in eq.(17). As shown in [10, 12], this constraint can be equivalently formulated as D^2 linear constraints.

Since Problem 2 is a standard SDP problem, it can be solved efficiently by many SDP algorithms. However, due to the relaxation in eq.(23), this SDP problem is just an approximation to the original LME problem. In problem 2, the optimization is carried out w.r.t. Z (which is constructed from all HMM Gaussian means) and ρ while A_{ij} and c_{ij} and Q are constants calculated from training data, initial models, and r is a pre-set parameter.

4. THE NEW COMPACT LME/SDP FORMULATION

In the above LME/SDP formulation, we construct a single variable matrix Z from all Gaussian means and size of Z is roughly proportional to square of total number of Gaussians in the model set. As a result, it leads to a very large scale SDP problem, especially for a large model set. Since a large-scale SDP problem is still relatively expensive to solve, in this work, we propose two novel methods to reformulate the LME problem. Compared with the above LME/SDP formulation, both methods result in much smaller SDP problems in number of free variables.

4.1. LME/SDP Formulation with Multiple Small Matrices

Instead of constructing a single variable matrix Z for all Gaussian mean vectors, we propose to construct many small variable matrices for all Gaussians in the model set. As a result, we will have an SDP problem with multiple small variable matrices. To elucidate the idea, we start from representing discriminant function $\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i})$ in matrix form. Starting from $\tilde{\boldsymbol{\mu}}_k$ in eq.(12) and $\tilde{\mathbf{x}}_{it}$ in eq.(15), we construct two matrices as follows:

$$Z_k := \begin{pmatrix} 1 & \tilde{\boldsymbol{\mu}}_k' \\ \tilde{\boldsymbol{\mu}}_k & \tilde{\boldsymbol{\mu}}_k\tilde{\boldsymbol{\mu}}_k' \end{pmatrix} \tag{25}$$

$$X_i^t := \frac{1}{2}\begin{pmatrix} \tilde{\mathbf{x}}_{it}'\tilde{\mathbf{x}}_{it} & -\tilde{\mathbf{x}}_{it}' \\ -\tilde{\mathbf{x}}_{it} & I_D \end{pmatrix}. \tag{26}$$

It is easy to verify that $X_i^t \cdot Z_{i_t} = \frac{1}{2}(\tilde{\mathbf{x}}_{it} - \tilde{\boldsymbol{\mu}}_{i_t})'(\tilde{\mathbf{x}}_{it} - \tilde{\boldsymbol{\mu}}_{i_t})$. Therefore, we can represent discriminant function $\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i})$ in the following matrix form:

$$\begin{aligned}\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) &= c_i - \frac{1}{2}\sum_{t=1}^{T}(\tilde{\mathbf{x}}_{it} - \tilde{\boldsymbol{\mu}}_{i_t})'(\tilde{\mathbf{x}}_{it} - \tilde{\boldsymbol{\mu}}_{i_t}) \\ &= c_i - \sum_{t=1}^{T} X_i^t \cdot Z_{i_t}. \end{aligned} \tag{27}$$

As the result, we can convert the constraint in eq.(7) into the following form:

$$\begin{aligned}\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_j) - \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) &= \sum_{t=1}^{T}\left[X_i^t \cdot Z_{i_t} - X_j^t \cdot Z_{j_t}\right] - c_{ij} \\ &= \sum_{k=1}^{\mathcal{K}} X_{ij}^k \cdot Z_k - c_{ij} \end{aligned} \tag{28}$$

where $X_{ij}^k = \sum_{t=1}^{T} \left[X_i^t \cdot \delta(i_t - k) - X_j^t \cdot \delta(j_t - k) \right]$ with Kronecker delta function $\delta(\cdot)$ and X_i^t and X_j^t calculated according to eq.(26) based on the Viterbi decoding paths **i** and **j** respectively.

Similarly, $R(\Lambda)$ in eq.(8) can be re-written using Z_k as follows:

$$\begin{aligned} R(\Lambda) &= \sum_{k=1}^{\mathcal{K}} (\tilde{\boldsymbol{\mu}}_k - \tilde{\boldsymbol{\mu}}_k^{(0)})'(\tilde{\boldsymbol{\mu}}_k - \tilde{\boldsymbol{\mu}}_k^{(0)}) \\ &= \sum_{k=1}^{\mathcal{K}} Q_k \cdot Z_k \le r^2 \end{aligned} \tag{29}$$

where Q_k is built as in eq.(26) using normalized initial mean vector $\tilde{\boldsymbol{\mu}}_k^{(0)}$ in place of $\tilde{\mathbf{x}}_{it}$.

In this way, we have a number of small variable matrices Z_k ($1 \le k \le \mathcal{K}$), each of which is constructed from a Gaussian mean vector $\tilde{\boldsymbol{\mu}}_k$ as in eq.(25). Obviously, rank of Z_k equals to one. Following [2], in order to transform it into an SDP problem, we relax the rank-one condition to a semidefinite condition for all Z_k as follows:

$$\text{rank}(Z_k) = 1 \quad \Longrightarrow \quad Z_k \succeq 0 \qquad k \in (1, 2, \cdots, \mathcal{K}). \tag{30}$$

Moreover, we must impose another constraint that the top-left element of Z_k equals to unity, i.e. $\{Z_k\}_{1,1} = 1$. This constraint can be easily cast as a linear constraint in matrix form.

Obviously, we can formulate the original LME problem as the following SDP problem after the relaxation in eq.(30):

Problem 3

$$\Lambda^* = \arg\min_{\Lambda} -\rho \tag{31}$$

subject to

$$\sum_{k}^{\mathcal{K}} X_{ij}^k \cdot Z_k + \rho \le c_{ij} \tag{32}$$

for all $X_i \in \mathcal{S}$ and $j \in \Omega$ but $j \ne W_i$.

$$\sum_{k=1}^{\mathcal{K}} Q_k \cdot Z_k \le r^2 \tag{33}$$

$$\rho \ge 0 \qquad Z_k \succeq 0 \qquad \text{for all } k \in (1, 2, \cdots, \mathcal{K}) \tag{34}$$

$$\{Z_k\}_{1,1} = 1 \qquad \text{for all } k \in (1, 2, \cdots, \mathcal{K}) \tag{35}$$

Apparently, Problem 3 is also an SDP problem, where optimization is performed with respect to all variable matrices Z_k and ρ. If we compare Problem 3 with Problem 2, their problem size varies significantly. As we know, problem size of an optimization problem is decided by the total number of free variables and the total number of constraints. For instance, the number of free variables (i.e., size of Z) in Problem 2 roughly equals to $(\mathcal{K} + D)(\mathcal{K} + D + 1)/2$. Meanwhile, we have $\mathcal{K}$ different matrices in Problem 3 and each one is only $(D+1)(D+2)/2$ in size, which amounts to $\mathcal{K}(D+1)(D+2)/2$ free variables in total. As we know, in most speech recognition systems, $\mathcal{K} \gg D$ always holds. Thus, number of free variables in Problem 3 is significantly less than that of Problem 2. Next, we investigate the number of constraints in Problem 2 and Problem 3. In addition to constraints imposed by support tokens and model, as in eqs. (21), (22), (32) and (33), which are the same for both problems, we only compare extra constraints imposed for structure of variable matrices, such as those in eqs.(24) and (35). Obviously, we have D^2 extra constraints in Problem 2 while there are $\mathcal{K}$ extra constraints in Problem 3. As the result, in Problem 3, we have largely reduced number of free variables but only slightly increased number of constraints. Overall, problem size of Problem 3 is much smaller than that of Problem 2, especially for large model sets.

4.2. LME/SDP Formulation with Decomposed Feature Vectors

From above, we can see that the key point to reduce problem size of SDP is to construct small variable matrices. In this section we propose another method to formulate LME as an even smaller SDP problem by decomposing feature vectors in construction of matrices.

As we know, in speech recognition, each D-dimension feature vector, $\mathbf{x}_{it}$, is normally composed of three equal-size parts, namely static feature $\bar{\mathbf{x}}_{it}$, delta feature $\dot{\mathbf{x}}_{it}$ and acceleration feature $\ddot{\mathbf{x}}_{it}$, i.e., $\mathbf{x}_{it} = (\bar{\mathbf{x}}_{it}; \dot{\mathbf{x}}_{it}; \ddot{\mathbf{x}}_{it})$. We can fold each feature vector $\mathbf{x}_{it}$ into a $\frac{D}{3} \times 3$ matrix as follows:

$$V_{it} := \begin{pmatrix} \bar{\mathbf{x}}_{it} & \dot{\mathbf{x}}_{it} & \ddot{\mathbf{x}}_{it} \end{pmatrix}_{\frac{D}{3} \times 3} \tag{36}$$

Next, we construct a matrix based on V_{it} as follows:

$$\widehat{X}_i^t := \begin{pmatrix} V_{it}'V_{it} & V_{it}' \\ V_{it} & I_{\frac{D}{3}} \end{pmatrix}. \tag{37}$$

Similarly, we can also fold each Gaussian mean vector, $\boldsymbol{\mu}_k$, into a matrix according to its static $\bar{\boldsymbol{\mu}}_k$, delta $\dot{\boldsymbol{\mu}}_k$ and acceleration $\ddot{\boldsymbol{\mu}}_k$ as follows:

$$U_k = \begin{pmatrix} \bar{\boldsymbol{\mu}}_k & \dot{\boldsymbol{\mu}}_k & \ddot{\boldsymbol{\mu}}_k \end{pmatrix}_{\frac{D}{3} \times 3} \tag{38}$$

In the same way, we construct a new variable matrix, $\widehat{Z}_k$ based on U_k as follows:

$$\widehat{Z}_k = \begin{pmatrix} I_3 & U_k' \\ U_k & U_k U_k' \end{pmatrix}. \tag{39}$$

Again, it is easy to verify that $\widehat{X}_i^t \cdot \widehat{Z}_{i_t} = \frac{1}{2}(\tilde{\mathbf{x}}_{it} - \tilde{\boldsymbol{\mu}}_{i_t})'(\tilde{\mathbf{x}}_{it} - \tilde{\boldsymbol{\mu}}_{i_t})$. Therefore, we have

$$\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) = c - \sum_{t=1}^{T} \widehat{X}_i^t \cdot \widehat{Z}_{i_t}. \tag{40}$$

Then, we can convert the constraint in eq.(7) into the following form:

$$\mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_j) - \mathcal{F}(\mathbf{X}_i|\boldsymbol{\lambda}_{W_i}) = \sum_{k=1}^{\mathcal{K}} \widehat{X}_{ij}^k \cdot \widehat{Z}_k - c_{ij} \tag{41}$$

where $\widehat{X}_{ij}^k = \sum_{t=1}^{T} \left[\widehat{X}_i^t \cdot \delta(i_t - k) - \widehat{X}_j^t \cdot \delta(j_t - k) \right]$ with $\widehat{X}_{rt}^i$ and $\widehat{X}_{rt}^j$ calculated according to eq.(37) based on the Viterbi decoding paths **i** and **j** respectively.

In the same way, $R(\Lambda)$ in eq.(8) can be re-written as follows:

$$R(\Lambda) = \sum_{k=1}^{\mathcal{K}} \widehat{Q}_k \cdot \widehat{Z}_k \le r^2 \tag{42}$$

where $\widehat{Q}_k$ is constructed as in eq.(37) by decomposing the initial Gaussian mean vector, $\tilde{\boldsymbol{\mu}}_k^{(0)}$, instead of feature vector.

Obviously, rank of each variable matrix, $\widehat{Z}_k$, constructed as in eq.(39), equals to three. Similarly, we relax the rank-three constraint in eq.(39) into a positive semidefinite constraint:

$$\text{rank}(\widehat{Z}_k) = 3 \quad \Longrightarrow \quad \widehat{Z}_k \succeq 0 \quad k \in (1, 2, \cdots, \mathcal{K}). \tag{43}$$

At last, the LME problem can be formulated as the following SDP problem after the relaxation in eq.(43):

Problem 4

$$\Lambda^* = \arg\min_{\Lambda,\rho} \quad -\rho \tag{44}$$

subject to

$$\sum_{k=1}^{\mathcal{K}} \widehat{X}_{ij}^k \cdot \widehat{Z}_k + \rho \leq c_{ij} \tag{45}$$

$$\text{for all } X_i \in \mathcal{S} \text{ and } j \in \Omega \; j \neq W_i.$$

$$\sum_{k=1}^{\mathcal{K}} \widehat{Q}_k \cdot \widehat{Z}_k \leq r^2 \tag{46}$$

$$\rho \geq 0 \qquad \widehat{Z}_k \succeq 0 \qquad \text{for all } \; k \in (1, 2, \cdots, \mathcal{K}) \tag{47}$$

$$\{\widehat{Z}_k\}_{1:3,1:3} = I_3 \qquad \text{for all } \; k \in (1, 2, \cdots, \mathcal{K}) \tag{48}$$

Obviously, in Problem 4, the total number of free variables has been further reduced since each variable matrix $\widehat{Z}_k$, $(\frac{D}{3}+3) \times (\frac{D}{3}+3)$ in size, is smaller than Z_k, $(D+1) \times (D+1)$ in size, in Problem 3.

5. EXPERIMENTS

The proposed compact LME/SDP methods are evaluated on the TIDIGITS database for continuous speech recognition in the string level[4]. The database consists of digits of '1' to '9', plus 'oh' and 'zero', for a total of 11 words. The length of digit strings varies from 1 to 7 (except 6). Only adult portion of the corpus is used in our experiments. It contains a total of 225 speakers (111 men and 114 women), 112 of which (55 men and 57 women) are used for training and 113 (56 men, 57 women) for test. The training set has 8623 digit strings and the test set has 8700 strings. Our model set consists of 11 whole-word CDHMMs representing all digits. Each HMM has 12 states and uses a simple left-to-right topology without state-skipping. Acoustic feature vectors consist of standard 39 dimensions (12 MFCC's and the normalized energy, plus their first and second order time derivatives). Different numbers of Gaussian mixture components (ranging from 1 to 32 Gaussians per state) are tested. We first train models based on maximum likelihood (ML) criterion. Then, the best ML model is used as seed model for MCE training. All HMM model parameters (except transition probabilities) are updated during the MCE training process. At last, we re-estimate the models with the LME method by using the previous SDP method as well as the newly proposed compact SDP approaches. In this work, we mainly compare complexity and optimization time of all different LME/SDP formulations. In LME, we use the best MCE model as the initial models and only HMM mean vectors are re-estimated. In each iteration of LME, a number of competing string-level models are computed for each utterance in the training set based on its N-best decoding results ($N = 5$). Then we select support tokens according to eq.(2) and obtain the optimal Viterbi sequence for each support token according to the recognition result. Then, the relaxed SDP optimization, e.g. Problems 2, 3, 4, are conducted with respect to variable matrices and ρ. At last, all HMM mean vectors are updated based on the optimization solution by projecting Z onto the appropriately constrained space of rank-one or rank-three matrices. If not convergent, next iteration starts again from recognizing all training data to generate N-Best competing strings. In our work, Problem 2, Problem 3 and Problem 4 are all solved by an open software, *DSDP v5.6* [1], running under Matlab.

Table 1. Problem sizes for all three different SDP formulations for the model set 32-mix, where SDP0 means our previous LME/SDP method in [11] and SDP1 denotes the LME/SDP formulation in section 4.1 and SDP2 stands for the LME/SDP formulation in section 4.2.

	SDP0	**SDP1**	**SDP2**
# variables	9,088,716	3,463,680	574,464
# constraints	1,521	4,224	38,016

5.1. Problem Size Comparison for Various LME/SDP Formulations

First of all, we will take the largest model set, i.e., 32-mix per state, as an example to compare problem sizes in number of total free variables and constraints. The model set 32-mix includes roughly 4,224 Gaussians. We have roughly estimated total number of free variables and constraints for three different SDP formulations in Table 1, where SDP0 denotes the previous LME/SDP formulation in section 3 originally proposed in [11], SDP1 denotes the new LME/SDP formulation with multiple small matrices in section 4.1 and SDP2 represents the compact LME/SDP formulation with decomposed feature vectors in section 4.2. From Table 1, it is clear that the new SDP formulations greatly reduce problem size of SDP, especially total number of free variables. For example, given the same model set, e.g. 32-mix per state, our previous formulation SDP0 forms an SDP problem consisting of about 9 million free variables. However, SDP1 reduces the total number of free variables to about 3 million (over 60% reduction) and SDP2 further decreases the number to about 0.6 million (about 94% relative reduction). Next, we also compare number of extra constraints related to structure of variable matrices for various LME/SDP formulations. From Table 1, we can see that SDP1 only needs slightly more constraints than SDP0 but the number of extra constraints significantly increases in SDP2 (about 30 times). On the whole, problem size of SDP1 and SDP2 has been largely reduced when compared with that of SDP0. Since optimization time and memory consumption of most SDP algorithms is related to a polynomial function of problem size, it is expected that optimization resources needed to solve SDP1 and SDP2 may be considerably reduced.

5.2. Performance and Optimization Time

Next, we investigate recognition performance of the two new LME/SDP formulations, namely SDP1 and SDP2. Both methods have been examined for different model sizes and their final recognition performance is compared with the previous approach SDP0 in [11] and the SOCP approach in [13]. The final results are summarized in Table 2. For reference, we also list recognition performance of baseline systems trained with ML and MCE criteria. From Table 2, we can clearly see that both SDP1 and SDP2 achieve good recognition performance especially for large model sets. Particularly, for the largest model set 32-mix, both new methods yield string error rate 0.51% (word error rate 0.17%), which is the best performance we have ever achieved on this task. It is slightly better than 0.53% string error rate obtained by SDP0. When we compare them with the SOCP method in [13], both SDP1 and SDP2 maintain good optimization accuracy due to the fact that the inherent SDP relaxation is much tighter than the SOCP relaxation used in [13]. This is particularly true for those large model sets such as 8-mix, 16-mix and 32-mix.

At the end, we compare optimization time needed by the opti-

Table 2. String error rates (in %) of various training methods on the TIDIGITS test data.

	ML	MCE	SDP0	SOCP	**SDP1**	**SDP2**
1-mix	12.61	6.72	2.75	2.10	2.67	2.56
2-mix	5.26	3.94	1.24	1.13	1.25	1.28
4-mix	3.48	2.23	0.89	0.98	0.94	0.92
8-mix	1.94	1.41	0.68	0.76	0.69	0.70
16-mix	1.72	1.11	0.63	0.66	0.63	0.64
32-mix	1.34	0.90	0.53	0.59	**0.51**	**0.51**

mizer software to solve all three SDP problems, i.e. SDP0, SDP1 and SDP2, and SOCP. The average optimization time per iteration (measured in second) are summarized in Table 3. From the results, we can see that the SOCP method still gives the best optimization efficiency, e.g. over hundreds times faster than SDP0 for the large model sets. Both SDP1 and SDP2 run slightly slower than SOCP but both methods show a huge improvement over SDP0 in terms of optimization time. For the large model sets, both methods run about tens of times faster than the previous SDP0 method because they have considerably reduced problem size of SDP. If we compare SDP1 with SDP2, we can see that SDP2 is about two times faster because SDP2 involves much less free variables even though it introduces more extra constraints.

Table 3. Average optimization time (in seconds) per iteration of various optimization methods on the TIDIGITS task. The numbers in parentheses indicate optimization speed ratio relative to SDP0.

	SDP0	SOCP	**SDP1**	**SDP2**
1-mix	1276	57 (x22)	511 (x2.5)	286 (x5)
2-mix	1068	58 (x18)	835 (x1.3)	372 (x3)
4-mix	3556	141 (x25)	1211 (x3)	483 (x8)
8-mix	12398	192 (x64)	1520 (x8)	758 (x16)
16-mix	40691	324 (x125)	2212 (x18)	1335 (x30)
32-mix	113110	506 (x223)	3173 (x36)	2343 (x48)

In summary, the newly proposed LME/SDP formulation (both SDP1 and SDP2) show a huge advantage in optimization efficiency compared with the previous SDP0 method and both SDP1 and SDP2 yield slightly better optimization accuracy and recognition performance than SDP0. On the other hand, both SDP1 and SDP2 are still slightly slower than the SOCP method but they can normally provide a much more precise solution and in turn yield better recognition performance.

6. CONCLUSIONS

In this work, we have proposed a new method to formulate large margin estimation (LME) of HMMs as smaller SDP problems. It can considerably reduce the total number of free variables in the resultant SDP problems and in turn significantly improve optimization efficiency in the SDP-based LME training, especially for large model sets. Meanwhile, the new methods still can achieve very good optimization accuracy since the problem is relaxed under the SDP framework, which is typically tighter than other types of convex relaxation, such as SOCP in [13]. The new methods open up a possible door to apply SDP-based LME to large vocabulary speech recognition tasks.

Acknowledgement

The authors acknowledge Mr. Xinwei Li for his helpful discussion on problem formulaiton and generous supports for building the baseline systems.

7. REFERENCES

[1] S. J. Benson, Y. Ye and X. Zhang, "Solving Large-Scale Sparse Semidefinite Programs for Combinatorial Optimization," *SIAM Journal on Optimization*, pp. 443-461, 10(2), 2000.

[2] P. Biswas, Y. Ye, "Semidefinite Programming for Ad Hoc Wireless Sensor Network Localization," *Proc. of 2004 Information Processing in Sensor Networks (IPSN)*, Berkeley, 46-54, Apr. 2004.

[3] S. Boyd, L. E. Ghaoui, E. Feron and V. Balakrishnan, "Linear matrix inequalities in system and control theory," *Society for Industrial and Applied Mathematics (SIAM)*, Philadelphia, PA, June 1994.

[4] H. Jiang, X. Li and C. Liu, "Large Margin Hidden Markov Models for Speech Recognition," *IEEE Trans. on Audio, Speech and Language Processing*, pp.1584-1595, Vol. 14, No. 5, September 2006.

[5] H. Jiang and X. Li , "Incorporating Training Errors For Large Margin HMMs Under Semi-definite Programming Framework," *Proc. of 2007 IEEE International Conference on Acoustic, Speech, and Signal Processing (ICASSP'2007)*, Hawaii, USA, April 2007.

[6] F. Sha and L. Saul, "Large Margin Gaussian Mixture Modeling for Phonetic Classification and Recognition," *Proc. of IEEE ICASSP 2006*, pp.265-268, Toulouse, France, 2006.

[7] F. Sha and L. Saul, "Large Margin Hidden Markov Models for Automatic Speech Recognition," *Advances in Neural Information Processing Systems (NIPS) 19*, 2006.

[8] X. Li, H. Jiang and C. Liu, "Large Margin HMMs for Speech Recognition," *Proc. of IEEE ICASSP'2005*, Pennsylvania, Mar. 2005.

[9] X. Li and H. Jiang, "A Constrained Joint Optimization Method for Large Margin HMM Estimation," *Proc. of 2005 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*,Nov. 2005.

[10] X. Li, "Large Margin Hidden Markov Models for Speech Recognition," *M.S. thesis*, Department of Computer Science and Engineering, York University, Canada, 2005.

[11] X. Li and H. Jiang, "Solving Large Margin Estimation of HMMs via Semidefinite Programming," *Proc. of 2006 International Conference on Spoken Language Processing (ICSLP'2006)*, Pittsburgh, USA, April 2006.

[12] X. Li and H. Jiang, "Solving Large Margin Hidden Markov Model Estimation via Semidefinite Programming," *to appear in IEEE Trans. on Audio, Speech and Language Processing*, 2007.

[13] Y. Yin and H. Jiang, "A Fast Optimization Method for Large Margin Estimation of HMMs based on Second Order Cone Programming," *Proc. of Interspeech 2007*, August 2007.

[14] Y. Yin, "A Study on Convex Relaxation for Large Margin Estimation of HMMs in Speech Recognition," *M.S. thesis*, Department of Computer Science and Engineering, York University, Canada, 2007.