Regularizing Linear Discriminant Analysis for Speech Recognition

Hakan Erdoğan

Faculty of Engineering and Natural Sciences
Sabanci University
Orhanli Tuzla 34956 Istanbul Turkey
haerdogan@sabanciuniv.edu

Abstract

Feature extraction is an essential first step in speech recognition applications. In addition to static features extracted from each frame of speech data, it is beneficial to use dynamic features (called Δ and $\Delta\Delta$ coefficients) that use information from neighboring frames. Linear Discriminant Analysis (LDA) followed by a diagonalizing maximum likelihood linear transform (MLLT) applied to spliced static MFCC features yields important performance gains as compared to MFCC+Δ+$\Delta\Delta$ features in most tasks. However, since LDA is obtained using statistical averages trained on limited data, it is reasonable to regularize LDA transform computation by using prior information and experience. In this paper, we regularize LDA and heteroschedastic LDA transforms using two methods: (1) Using statistical priors for the transform in a MAP formulation (2) Using structural constraints on the transform. As prior, we use a transform that computes static+Δ+$\Delta\Delta$ coefficients. Our structural constraint is in the form of a block structured LDA transform where each block acts on the same cepstral parameters across frames. The second approach suggests using new coefficients for static, first difference and second difference operators as compared to the standard ones to improve performance. We test the new algorithms on two different tasks, namely TIMIT phone recognition and AURORA2 digit sequence recognition in noise. We obtain consistent improvement in our experiments as compared to MFCC features. In addition, we obtain encouraging results in some AURORA2 tests as compared to LDA+MLLT features.

1. Introduction

One of the main components in a pattern recognition system is the feature extractor. Feature extraction is an important step for speech recognition since the time-domain speech signal is highly variable, thus complex linear and nonlinear processing is required to obtain low-dimensional and reasonably less variant features. Speech signal is partitioned in time into overlapping frames and static features are obtained by processing each frame separately. It has been known that using dynamic features is also useful in speech recognition [1]. These dynamic features, called Δ and $\Delta\Delta$ features, are obtained by taking first and second difference of static features in a neighborhood around the current frame.

Linear discriminant analysis (LDA) applied to spliced static features attempts to find a transform that will extract the dynamic information from the neighboring static features automatically. The criterion is to choose transformed dimensions that retain most of the discriminating information. LDA assumes each class has the same within class covariance. Heteroscedastic LDA [2] enables one to model each class with a different within class covariance matrix. LDA transform works best when a diagonalizing maximum likelihood linear transform (MLLT) [3] is applied afterwards to rotate the axes so that the classes have more diagonal covariances. For (diagonal) HLDA, MLLT is not required since it attempts to solve both dimension reduction and diagonalization problems in a single step [2].

In this paper, we introduce regularization methods for LDA and heteroscedastic LDA (HLDA) transforms. We have developed two methods. One is based on a Bayesian framework for transform coefficients. The second one is based on constraining the LDA transform structure. In section 2, we introduce and derive the solution for the Bayesian HLDA method. We describe the block structured LDA method in section 3. In section 4, we present our experimental results on TIMIT and AURORA2 databases. Finally, we state our conclusions in section 5.

2. Bayesian HLDA

Linear discriminant analysis is performed by maximizing Fisher's discriminant[4]. The solution can also equivalently be found using a maximum likelihood formulation [2] assuming Gaussian distributions for classes. We seek to find a square matrix $\boldsymbol{A}$ to be applied to the feature vectors such that all the discriminatory information is retained in the first p dimensions after transformation. This is formally achieved by requiring that the last $n-p$ dimensions of the transformed features share the same mean vector and covariance matrix across all classes [2]. When all the classes are assumed to share the same within class covariance matrix in the transformed space, we obtain the LDA result. By allowing each class to have its separate diagonal covariance matrix, we arrive at the heteroscedastic LDA transform [2]. We review the HLDA derivation below.

Let $\boldsymbol{x}_i \in \mathbb{R}^n : i = 1 \ldots N$ be feature vectors in the original space. Furthermore, each $\boldsymbol{x}_i$ is labeled with a class $c_i = j \in 1, \ldots, J$ We would like to find a transformation $\boldsymbol{y} = \boldsymbol{A}_p \boldsymbol{x}$, $\boldsymbol{A}_p : \mathbb{R}^n \rightarrow \mathbb{R}^p$ with $p < n$. We seek to choose new features $\boldsymbol{y}$ such that most of the class-discriminating information in $\boldsymbol{x}$ is retained in $\boldsymbol{y}$. For maximum likelihood formulation, we stack $\boldsymbol{A}_{n-p}$ which has $n-p$ rows to the transformation $\boldsymbol{A}_p$ to form the transform

$$\boldsymbol{A} = \left[\begin{array}{c} \boldsymbol{A}_p \\ \boldsymbol{A}_{n-p} \end{array} \right].$$

We require diagonal covariance Gaussian models for transformed classes and furthermore last $n-p$ dimensions for each class share the same mean and covariance matrix. We then find $\boldsymbol{A}$ that maximizes likelihood of the training data under these

modeling constraints. The likelihood of the training data as a function of $\boldsymbol{A}$ can be written as follows [2, 5]:

$$L(\boldsymbol{A}) = \sum_{j=1}^{J} \frac{N_j}{2N} \log \frac{|\boldsymbol{A}|^2}{|\text{diag}(\boldsymbol{A}_p \boldsymbol{W}_j \boldsymbol{A}_p)||\text{diag}(\boldsymbol{A}_{n-p} \boldsymbol{T} \boldsymbol{A}_{n-p})|}$$

where

$$\boldsymbol{W}_j = \frac{1}{N_j} \sum_{c_i=j} (\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T$$

are the estimated within class covariances and

$$\boldsymbol{T} = \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T$$

is the estimated total covariance of the training data. Here $\boldsymbol{\mu}_j$ are class means and $\boldsymbol{\mu}$ is the overall mean.

Direct maximization of the likelihood function is not possible and we have to use iterative techniques. Since the likelihood is not convex, iterative methods can be tricky to implement as well. In [2], a steepest descent algorithm is used, however Gales provides a faster row-update algorithm in [5]. We rewrite the likelihood function using rows of $\boldsymbol{A}$ to arrive at that derivation:

$$\begin{aligned} L(\boldsymbol{A}) &= \log(\boldsymbol{a}_r^T \boldsymbol{c}_r) - \frac{1}{2} \sum_{r=p+1}^{n} \log \boldsymbol{a}_r^T \boldsymbol{T} \boldsymbol{a}_r \qquad (1) \\ &\quad - \frac{1}{2} \sum_{j=1}^{J} \frac{N_j}{N} \sum_{r=1}^{p} \log \boldsymbol{a}_r^T \boldsymbol{W}_j \boldsymbol{a}_r, \end{aligned}$$

where $\boldsymbol{a}_r^T$ is the rth row of $\boldsymbol{A}$ [1] and $\boldsymbol{c}_r^T$ is the cofactor of $\boldsymbol{a}_r^T$. Note that the first term can be written using any row r.

Our goal in this section is to derive a Bayesian estimation formula for $\boldsymbol{A}$ where we assume there is a prior distribution of the matrix entries in $\boldsymbol{A}$. For simplicity, we assume a diagonal covariance Gaussian prior for the HLDA matrix $\boldsymbol{A}$. We can write the aposteriori objective function as follows:

$$\Phi(\boldsymbol{A}) = -L(\boldsymbol{A}) + 1/2 \sum_{r=1}^{n} (\boldsymbol{a}_r - \bar{\boldsymbol{a}}_r)^T \boldsymbol{P}_r (\boldsymbol{a}_r - \bar{\boldsymbol{a}}_r),$$

where $\bar{\boldsymbol{a}}_r$ is the mean vector for row $\boldsymbol{a}_r$ and $\boldsymbol{P}_r$ is the precision matrix (inverse covariance). This objective needs to be minimized.

The gradient of the objective function with respect to $\boldsymbol{a}_r$ can be computed easily as:

$$\begin{aligned} \nabla_{\boldsymbol{a}_r} \Phi &= \boldsymbol{P}_r (\boldsymbol{a}_r - \bar{\boldsymbol{a}}_r) - \frac{\boldsymbol{c}_r}{\boldsymbol{a}_r^T \boldsymbol{c}_r} \qquad (2) \\ &\quad + \begin{cases} \sum_{j=1}^{J} \frac{N_j}{N} \frac{\boldsymbol{W}_j \boldsymbol{a}_r}{\boldsymbol{a}_r^T \boldsymbol{W}_j \boldsymbol{a}_r} & r \le p \\ \frac{\boldsymbol{T} \boldsymbol{a}_r}{\boldsymbol{a}_r^T \boldsymbol{T} \boldsymbol{a}_r} & r > p \end{cases}. \end{aligned}$$

To solve the minimization problem, we need to set $\nabla_{\boldsymbol{a}_r} \Phi = 0$ and solve for each $\boldsymbol{a}_r$. This appears to be untractable without using iterative methods. We use a trick similar to the one in [5] and assume that $\boldsymbol{a}_r^T \boldsymbol{c}_r$, $\boldsymbol{a}_r^T \boldsymbol{W}_j \boldsymbol{a}_r$ and $\boldsymbol{a}_r^T \boldsymbol{T} \boldsymbol{a}_r$ are quantities that do not change much from iteration to iteration and plug in their previous values in the equation and solve for $\boldsymbol{a}_r$ easily in $\nabla_{\boldsymbol{a}_r} \Phi = 0$. This yields the following simple algorithm.

[1] All vectors are column vectors

Start with $\boldsymbol{A} = \bar{\boldsymbol{A}}$
while not converged
  for each $r = 1, \ldots, n$
    Compute $\boldsymbol{G}_r = \begin{cases} \sum_{j=1}^{J} \frac{N_j}{N} \frac{\boldsymbol{W}_j}{\boldsymbol{a}_r^T \boldsymbol{W}_j \boldsymbol{a}_r} & r \le p \\ \frac{\boldsymbol{T}}{\boldsymbol{a}_r^T \boldsymbol{T} \boldsymbol{a}_r} & r > p \end{cases}$
    Compute $\alpha_r = (\boldsymbol{a}_r^T \boldsymbol{c}_r)^{-1} = |\boldsymbol{A}|^{-1}$
    Update $\boldsymbol{a}_r = (\boldsymbol{G}_r + \boldsymbol{P}_r)^{-1} (\alpha_r \boldsymbol{c}_r + \boldsymbol{P}_r \bar{\boldsymbol{a}}_r)$
  end
end

Note that this approximation is somehow different than the one in [5], however this yields $\boldsymbol{a}_r$ that are in the same direction as the one in [5] when there is no prior ($\boldsymbol{P}_r = 0$). This is acceptable since scaling rows of $\boldsymbol{A}$ do not change the maximum likelihood objective function [2].

When the within class covariances $\boldsymbol{W}_j$ are assumed to be equal and when the common $\boldsymbol{W}$ is estimated as the weighted average of within class covariances, we obtain the regular LDA solution using the above objective function [2]. The update equations are flexible to allow for HLDA solution when different $\boldsymbol{W}_j$ are used (with $\boldsymbol{P}_r = 0$). In its most general form, we can use a prior for the transform matrix $\boldsymbol{A}$ and optimize the MAP objective function using the algorithm above.

Usually, we will have a prior mean which is a $p \times n$ matrix. In that case, we can set $\boldsymbol{P}_r = 0$ for $r > p$ so that no prior is used for rows greater than p. For rows $r \le p$, we use the same precision matrix which is a scaled identity matrix $\boldsymbol{P}_r = \beta \boldsymbol{I}$. One could also experiment with different precision matrices.

As prior mean transform, there are many possibilities. We have used the Static+Δ+$\Delta\Delta$ (S+D+DD) transform as the prior mean in this study. When the static features are thirteen dimensional and a neighborhood of seven frames are spliced together, and when for example HTK 3.2 is used with parameters DELTAWINDOW=2 and ACCWINDOW=1, this amounts to the transform (ignoring scaling of each row)

$$\bar{\boldsymbol{A}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -2 & -1 & 0 & 1 & 2 & 0 \\ 2 & 1 & -2 & -2 & -2 & 1 & 2 \end{bmatrix} \otimes \boldsymbol{I}_{13 \times 13}$$

Here $\otimes$ denotes the Kronecker product. We know that this transform yields reasonable results, so it makes sense use this transform as a prior mean. However there could be other choices such as using an LDA transform computed from a smaller neighborhood of frames and extended to the larger neighborhood by inserting zeroes in it. This would make sure that the new transform does not deviate too much from the earlier, smaller but possibly more reliable, transform.

A disadvantage of using a Gaussian prior is that the transform computed will not be sparse even if we start with a sparse transform as the one given above. To keep sparsity, for example a Laplacian prior would work better. Another approach to keep sparsity is by constraining the LDA transform to have a certain structure, such as block structured, which we explore next.

3. Block structured LDA

Intuitively, enforcing a structure on the linear discriminant transform matrix $\boldsymbol{A}_p$ would be a good regularization technique. For example, we could tie coefficients in matrix $\boldsymbol{A}_p$ either to other coefficients or to fixed values (mostly zero). Another option is to enforce a block structure on the linear transform $\boldsymbol{A}_p$.

In this paper, we only enforce a simple block structure, that is we divide the original dimensions into groups and compute a dimension reducing transform for each group. This amounts to setting $\boldsymbol{A}_p$ coordinates to zero for dimensions that are not in the group. This structure is similar to the S+D+DD transform introduced in the previous section, but would allow using different coefficients for neighboring frames.

Implementation of such a transform is trivial. We determine dimension groups and decide on how many lower dimensions to reduce them to. Then, we use the corresponding rows and columns of $\boldsymbol{W}$ and $\boldsymbol{T}$ to compute a lower dimensional LDA transform to achive the result. We then stack each lower dimensional transform to obtain $\boldsymbol{A}_p$.

Typically, we choose a dimension group to contain the same cepstral parameters in every frame in the spliced vector. Thus, in a thirteen static parameter, seven spliced frame scenario, we would be using thirteen groups of seven dimensions each. We will be reducing each seven dimensional group to three dimensions which will yield in the end a 39×91 dimensional $\boldsymbol{A}_p$ matrix which is highly sparse. This clearly is a way to replace static, first difference and second difference operator coefficients for each cepstral parameter. It turns out the estimated coefficients indeed can be easily identified as being similar to the static, first difference or second difference coefficients as we show in the experimental results section.

4. Experimental Results

We applied the introduced methods to two different databases: TIMIT and AURORA2.

We performed phone recognition experiments on TIMIT database [6]. We mapped 64 phones in TIMIT transcriptions down to 48 as in [6] for obtaining monophone models. During performance calculations, we further mapped 48 phones down to 39 as is typical[6].

We built tied state triphone models using different features with 39 dimensions each. MFCC features are standard 12 cepstra + energy and Δ and $\Delta\Delta$ dynamic features. LDA type features are obtained by transforming 91 dimensional spliced static MFCC features from 7 neighboring frames to 39 dimensions and applying MLLT transform afterwards. In HLDA-MAP+MLLT method, S+D+DD transform is used as the prior mean and $\boldsymbol{P}_r = 1000\boldsymbol{I}$ for $r \leq p$. We used blocks similar to the S+D+DD transform in the block LDA method as mentioned earlier. We define each tied HMM state as a class and obtain statistics $\boldsymbol{W}_j$ and $\boldsymbol{T}$ using Viterbi aligned training data.

The results are tabulated in Table 1. We obtained the best result using LDA+MLLT features. HLDA method should be better than LDA theoretically when the covariances are known exactly. However, since we estimate covariance matrices from limited amount of data, we suspect that HLDA uses unreliable $\boldsymbol{W}_j$ estimates and performs worse in testing. We have tested classification performance of HLDA and LDA+MLLT using Monte Carlo simulations and verified that indeed when the covariances are known exactly, HLDA classifies better than LDA+MLLT. We believe this result did not follow in the real case due to unreliable covariance estimates $\boldsymbol{W}_j$. More reliable estimates of within class covariances could be obtained by smoothing as in regularized discriminant analysis of Friedman [7] which is also applied to speech recognition recently [8]. The $\boldsymbol{W}$ matrix used in LDA is more reliable since it is obtained by averaging much more data. The Bayesian and block structured approaches also cannot surpass LDA+MLLT performance although they perform better than the MFCC baseline. We attribute these results to the fact that there is no noise and channel mismatch between training and test data and it appears no regularization is needed to improve LDA+MLLT result.

Features	Accuracy	Correct Detection
MFCC	62.92%	80.34%
LDA+MLLT	**70.59%**	**82.34%**
HLDA	68.57%	81.49%
HLDA-MAP+MLLT	69.25%	82.07%
Block LDA	68.05%	80.46%

Table 1: Phone recognition accuracy and correct detection rates on TIMIT test set with different features of dimension 39.

AURORA2 is a standard database distributed by ETSI which can be used for testing noise robust recognition algorithms. The task is to detect digit sequences (from TIDIGITS database) when different types and amounts of noise is added to the utterances. ETSI has published an advanced distributed speech recognition front-end (ES 202 050) that achieves very good performace as compared to MFCC features under noise. They are obtained by two-stage Wiener filtering of speech data before extracting MFCC features from the preprocessed speech. We call these features AFE features. We have performed experiments on AURORA2 database using the clean training data only.

In Table 2, we show recognition accuracy results under differing noise conditions with various features. MFCC denotes MFCC features without any preprocessing, AFE is the advanced front-end features which improves significantly using intelligent preprocessing of speech data. We have based our LDA type features on AFE features. Each state is considered as a class similar to the TIMIT experiment. Once again, LDA+MLLT features improve upon AFE features quite significantly. HLDA performs much worse as compared to LDA+MLLT, we conjecture once again that the within class covariances are not robust enough. For low SNR conditions, the regularized block LDA performs better than all other methods. This shows that the block LDA method is more robust to modeling mismatches.

Finally, in Figure 1, we plot seven primary coefficients for each cepstral parameter (on top of each other) that will multiply each frame in the spliced vector obtained using the block LDA method. As we can observe from the plot, the primary LDA row for each cepstral parameter was found to be kind of a weighted averager. Thus, block LDA replaces the static feature with an averaged static feature. Seven secondary coefficients are plotted in Figure 2. For eleven cepstral coefficients, these act similar to a first difference operator, for other two coefficients, they act similar to a second difference operator. The tertiary coefficients (not shown) act as second difference for the previous eleven coefficients and as first difference for the remaining two.

5. Conclusion

LDA+MLLT method works well for the two different domains that we experimented with. We were able to outperform MFCC+D+DD features using LDA+MLLT in both test sets. Furthermore, applying LDA+MLLT on top of ETSI advanced front end features yields consistent improvement in Aurora2 tests. Our attempts at regularizing the LDA transform were promising but not consistently better than the unregularized case. We only obtained limited improvement for the AURORA2 task using the block LDA method in extremely noisy

Features/SNR(dB)	clean	20	15	10	5	0	-5
MFCC	99.0	94.1	85.0	65.5	38.6	17.1	8.5
AFE-MFCC	99.1	98.0	96.5	92.4	82.3	58.2	27.2
LDA+MLLT	**99.3**	**98.3**	**97.1**	**93.4**	**83.1**	58.7	27.2
HLDA	98.4	96.6	94.4	85.4	60.4	27.8	12.1
HLDA-MAP+MLLT	**99.3**	98.2	96.8	92.8	81.4	54.9	22.3
Block LDA	99.2	97.9	96.6	92.8	83.0	**60.3**	**28.8**

Table 2: AURORA2 clean training speech recognition accuracy rates under different SNRs using various features of dimension 39. The results are averaged over all test sets A, B and C containing all ten noise types. Total reference word count is 32883 for each SNR type.

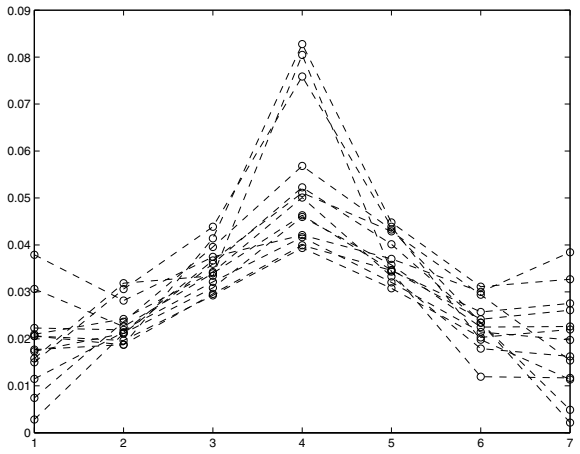

Figure 1: Primary "block structured LDA" coefficients for 13 cepstral coefficients

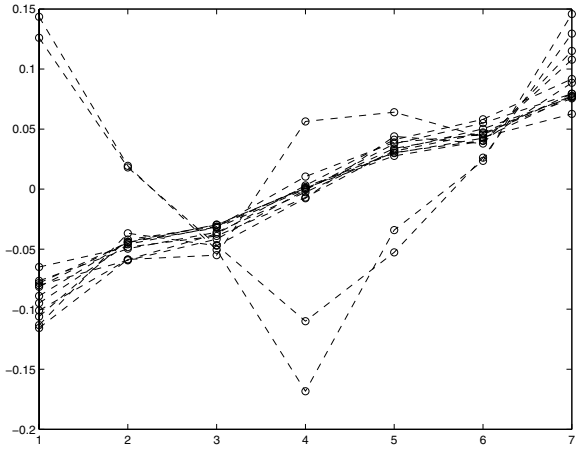

Figure 2: Secondary "block structured LDA" coefficients for 13 cepstral coefficients

conditions.

Block LDA method appears to compute static, first difference and second difference feature parameters by optimally determining the weights of cepstral coefficients across frames. The weights are obtained by maximizing the Fisher's discriminant. This method is a fast and straightforward way to optimize them. In both our experiment domains, we obtain consistent improvement by using block LDA method over using standard weights.

The HLDA-MAP method is a generalization of the regular HLDA and LDA methods, however it is not easy to determine appropriate prior parameters for optimal performance. We believe the suboptimal results we obtained can be improved by using more appropriate prior parameters and improved within-class covariance estimates via smoothing [8]. The prior parameters could be task dependent as well. Further investigation into regularization methods and parameter estimation is required to improve the performance of the dimension reducing discriminative transforms for speech recognition.

6. References

[1] S. Furui, "Speaker independent isolated word recognition using dynamic features of speech spectrum," *IEEE Tr. Acoust. Sp. Sig. Proc.*, vol. 34, no. 1, pp. 52–59, 1986.

[2] N. Kumar and A. G. Andreou, "Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition," *Speech Communication*, vol. 26, pp. 283–97, 1998.

[3] R. A. Gopinath, "Maximum likelihood modeling with Gaussian distributions for classification," in *Proc. IEEE Conf. Acoust. Speech Sig. Proc.*, volume 2, pp. 661–4, 1998.

[4] R. O. Duda and P. E. Hart, *Pattern classification and scene analysis*, John Wiley & Sons, New York, 1973.

[5] M. J. F. Gales, "Semi-tied covariance matrices for hidden Markov models," *IEEE Tr. Speech and Audio Proc.*, vol. 7, no. 3, pp. 272–81, May 1999.

[6] K.-F. Lee and H.-W. Hon, "Speaker-independent phone recognition using hidden Markov models," *IEEE Tr. Acoust. Sp. Sig. Proc.*, vol. 37, no. 11, pp. 1641–1648, November 1989.

[7] J. H. Friedman, "Regularized discriminant analysis," *J. Amer. Statistical Assoc.*, no. 84, pp. 165, 1989.

[8] L. Burget, "Combination of speech features using smoothed heteroscedastic linear discriminant analysis," in *Proc. IEEE Conf. Acoust. Speech Sig. Proc.*, 2005.