Multiple Linear Transformations for Classification

Ramesh A. Gopinath
IBM T. J. Watson Research, Yorktown Heights, NY
http://www.research.ibm.com/people/r/rameshg/

Johns Hopkins University Presentation, July 18, 2001

Joint work with Nagendra Goel, LSI Logic, Gaithersburg
• Classification Problem: Given labeled training data

\[ \{(x_t, c_t)\}, \quad x_t \in \mathbb{R}^n, \quad c_t \in \mathcal{C} = \{1, 2, \ldots, C\}, \quad t = 1, 2, \ldots, T. \]

learn a classifier \( f : \mathbb{R}^n \rightarrow \mathcal{C} \).

• Statistical Models:

  – Generative Model: Class-conditionals, \( p(x|c) \), and priors

    \[ \hat{f}(x) = \arg \max_c p(x|c)p(c) \]

  – Discriminative Model: Posterior \( p(c|x) \), e.g., logistic regression

    \[ \hat{f}(x) = \arg \max_c p(c|x). \]

• Other Approaches: e.g., Support Vector Machines that learn
Gaussian Mixture Model

- Gaussian Mixture Model (GMM):
  \[
  p(x|c) = \sum_{k=1}^{K^c} \pi_k^c N(x; \mu_k^c, \Sigma_k^c).
  \]

- Parameters: Priors, Means and Covariances of the Gaussians
  \[
  x \sim \{\pi_j, \mu_j, \Sigma_j\}, \quad j = 1, 2, \ldots, J.
  \]

- Practical considerations (computational, storage and data-sparsity issues - esp. when \(J\) and \(n\) are large) lead to constrained estimation of the parameters
  - e.g., \(\Sigma_j\) is diagonal, say, \(D_j\).

- Constrained GMM may not be Invariant to Linear Transformations (ILT) of the data; in which case one hopes to better satisfy constraints by linearly transforming the data.
• MLLT: Diagonal Covariance GMM is not ILT. If $y = Ax$, $A \in \mathbb{R}^{n \times n}$ is modeled by a Diagonal Covariance GMM,

$$Ax = y \sim \{\pi_j, \mu_j, D_j\},$$

then, $x$ is also a GMM with parameters of the form

$$A^{-1}y = x \sim \{\pi_j, A^{-1}\mu_j, A^{-1}D_jA^{-T}\}.$$

• ML Estimation of MLLT Model Parameters - Generalized EM
  – Fix all other parameters and estimate $A$ - Non-linear Programming (or row-by-row update a la Mark Gales).
  – Fix $A$ and estimate $\{\pi_j, \mu_j, D_j\}$ - Standard EM algorithm.

• Improved classification accuracy with no computational overhead.
Multiple Linear Transforms

- Let $A \in \mathbb{R}^{N \times n}$, $n \leq N \leq nj$ and let $y = Ax \in \mathbb{R}^N$. Each Gaussian has an index set $S_j$ that specifies precisely $n$ distinct rows of $A$.
- For Gaussian $j$, let $y_j = A_j x$, be the sub-vector of $y$ corresponding to $S_j$.
- Each $y_j$ is modeled as a Diagonal Covariance Gaussian
  $$y_j \sim \{\pi_j, \mu_j, D_j\}.$$ Equivalently $x$ is modeled by a GMM with parameters of the form $$A_j^{-1} y_j = x \sim \{\pi_j, A_j^{-1} \mu_j, A_j^{-1} D_j A_j^{-T}\}.$$ 
- e.g.,
  $$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}, \ S_1 = \{1, 2\}, \ S_2 = \{1, 3\}, \ A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$
Multiple Linear Transforms

- MLT Parameters: \( \{A, S_j, \pi_j, \mu_j, D_j\} \).

- Parameter Estimation - EM Algorithm
  
  - E-Step: Compute Gaussian posteriors \( \gamma_j(t) \) for sample
  
  - M-step: Maximize wrt \( \{A, S_j, \pi_j, \mu_j, D_j\} \)

\[
\sum_{j,t} \gamma_j(t) \left[ 2 \log \pi_j + \log |A_j^T D_j^{-1} A_j| - (x_t - A_j^{-1} \mu_j)^T A_j^T D_j^{-1} A_j \right]
\]

- Estimation of \( \{\pi_j, \mu_j, D_j\} \): If \( \bar{\pi}_j, \bar{\mu}_j \) and \( \bar{\Sigma}_j \) are the sample means and sample covariances respectively

\[
\gamma_j = \sum_t \gamma_j(t), \quad \bar{\pi}_j = \frac{\gamma_j}{T}, \quad \bar{\mu}_j = \sum_t \frac{\gamma_j(t)}{\gamma_j} x_t, \quad \bar{\Sigma}_j = \sum_t \frac{\gamma_j(t)}{\gamma_j} (x_t - \bar{\mu}_j)(x_t - \bar{\mu}_j)^T
\]

then the optimal values of \( \{\pi_j, \mu_j, D_j\} \) for fixed \( \{A, S_j\} \) are

\[
\pi_j = \bar{\pi}_j, \quad \mu_j = A_j \bar{\mu}_j, \quad D_j = diag(A_j \bar{\Sigma}_j A_j^T)
\]
• For fixed \((A, S_j)\) the ML value is proportional to (using optimal values of \(\{\pi_j, \mu_j, D_j\}\) as functions of \((A, S_j)\))

\[
L(A, S_j) = \sum_j \gamma_j \left[ \log |A_j| - \frac{1}{2} \log |\text{diag}(A_j \tilde{\Sigma}_j A_j^T)| \right]
\]

• \(L(A, S_j)\) can be maximized wrt \(A\) using any standard non-linear optimizer (or using a variant of the row-by-row technique).

• Maximizing \(L(A, S_j)\) wrt \(S_j\) is difficult - a discrete optimization problem. We propose two heuristic methods to solve this problem.

• Computational Cost: But for premultiplication of \(x\) by \(A\) (large if \(N >> n\)) the same cost (and code!!) as implementing a Diagonal Covariance GMM.
Choosing $S_j$ - Bottom-Up Clustering Approach

1. Initialization: Start with $A \in \mathbb{R}^{nJ \times n}$ and $S_j = \{n(j-1) + 1, \ldots, nj\}$.

   $A = \begin{bmatrix} U_1 \\ \vdots \\ U_J \end{bmatrix}$ where the rows of $U_j$ are eigenvectors of $\bar{\Sigma}_j$.

2. Pairs of rows, say $(r, s)$, are merge candidates if the pair does not co-occur in any $S_j$. A merge of the candidate pair $(r, s)$ is defined as follows: drop the $s^{th}$ row of $A$ and replace $s$ with $r$ in $S_j$.

3. Find and merge the best pair - i.e., the candidate merge pair with least loss in likelihood.

4. Maximize $L(A, S_j)$ with respect to $A \in \mathbb{R}^{k \times n}$.

5. If $(k > N)$ go to 3.
Choosing $S_j$ - Heuristic Top-down Splitting Algorithm

1. Initialization: Start with $A \in \mathbb{R}^{n \times n}$ and $S_j = \{1, 2, \ldots, n\}$ initialized with MLLT.

2. Randomly pick the $r^{th}$ row of $A \in \mathbb{R}^{k \times n}$, say $a$. Let
   
   $$A \leftarrow \begin{bmatrix} A \\ a \end{bmatrix} \in \mathbb{R}^{k+1 \times n}.$$

3. For all $j$ such that $r \in S_j$ randomly replace $r$ with $k + 1$ or $r$.


5. For all $j$ such that $r \in S_j$ replace $r$ with the row that maximizes the likelihood.

6. Maximize $L(A, S_j)$ wrt $A$. Repeat 5 and 6 several times.

7. If $k < N$ go to 2.
- Command and Control Speech Recognition Task in a Car.
- Two microphone positions - Seatbelt and Sun Visor.
- Acoustic Model - 3644 Gaussians, 809 HMM states (or classes).

<table>
<thead>
<tr>
<th>Model</th>
<th>Seat Belt</th>
<th>Sun Visor</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLLT</td>
<td>6.29</td>
<td>6.37</td>
</tr>
<tr>
<td>MLT(N=156)</td>
<td>6.06</td>
<td>6.22</td>
</tr>
<tr>
<td>FullCovariance</td>
<td>4.91</td>
<td>5.15</td>
</tr>
</tbody>
</table>

Table 1: String Error Rates for the MLLT, MLT and Full Covariance Models
MLT Experimental Results - II

- 7/11 Digit String Recognition Task in a Noisy Car.
- Acoustic Model - 5261 Gaussians, 809 HMM states (or classes).

<table>
<thead>
<tr>
<th>Model</th>
<th>Seat Belt Microphone</th>
<th>Sun Visor Microphone</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 mph</td>
<td>30 mph</td>
</tr>
<tr>
<td>MLLT</td>
<td>5.68</td>
<td>5.25</td>
</tr>
<tr>
<td>MLT (N=156)</td>
<td>5.14</td>
<td>4.33</td>
</tr>
<tr>
<td>Full Covariance</td>
<td>4.76</td>
<td>4.03</td>
</tr>
</tbody>
</table>

Table 2: String error rates for MLLT, MLT and Full Covariance Models at various noise conditions
Ongoing Work - An Extension of MLLT - EMLLT

- **MLLT:**
  \[ \Sigma_j^{-1} = A^T D_j^{-1} A. \]

- **MLT:**
  \[ \Sigma_j^{-1} = A_j^T D_j^{-1} A_j = A^T \Lambda_j A, \text{ diagonal } \Lambda_j \in \mathbb{R}_+^{N \times N}, \Lambda_j(S_j) \]

- **EMLLT\textsuperscript{a}:**
  \[ \Sigma_j^{-1} = A^T \Lambda_j A, \text{ diagonal } \Lambda_j \in \mathbb{R}^{N \times N}, A^T \Lambda_j A > 0. \]

Parameters are \( \{A, \pi_j, \mu_j, \Lambda_j\} \). No discrete optimization - training easier than MLT. EMLLT computationally more expensive at test time.

- EMLLT gives 20% relative improvement over MLT. EMLLT performs 15% better than full-covariance modeling!

\textsuperscript{a}joint work with Peder Olsen, Submitted to NIPS 2001 & Trans. SAP
• MLT is a simple technique to get close to a full-covariance classification performance while retaining the advantage of diagonal covariance GMMs.

• Training MLT requires (over and above the standard estimation in diagonal covariance GMMs) estimation of the MLT matrix $A$ and assignments of rows of $A$ to Gaussians - i.e., $S_j$.

• $A$ can be estimated using numerical techniques. $S_j$ is hard. Two heuristics were proposed to solve this - top-down splitting MLLT and bottom-up clustering of eigenvectors of $\bar{\Sigma}_j$.

• Further enhancements to MLT are possible, e.g., EMLLT.
Enhancing GMM Scores Using SVM Hints

Ramesh A. Gopinath
IBM T. J. Watson Research, Yorktown Heights, NY 10598
http://www.research.ibm.com/people/r/rameshg/

Johns Hopkins University Presentation, July 18, 2001

Joint work with Shai Fine and Jiri Navratil
• SVM is an **Optimal Linear Classifier** for two-class problems, i.e., maximal margin linear classifier with respect to the training data.

\[ \{ x \mid (w \cdot x) + b = -1 \} \]

\[ \{ x \mid (w \cdot x) + b = +1 \} \]

\[ \{ x \mid (w \cdot x) + b = 0 \} \]

\[ y_i = -1 \]

\[ y_i = +1 \]

• Soft-margin approach for non-separable training data.

• **Kernel Trick** - Allows implicit non-linear transformation of data into a high-dimensional space and builds optimal linear classifier there. All computations are in the original feature space i.e., efficient.
SVM: Advantages and Disadvantages

- Good generalization (Margin maximization)
- Global convergence (Quadratic Programming Problem)
- State-of-the-art performance in OCR, Document classification, Bioinformatics, Speech, etc.
- Kernel selection (Fisher-Kernel, feature $\phi(x) = \nabla_\theta \log p_\theta(x)$)
- Handling multi-class problems (Error-Correcting Output Codes (ECOC))
- Handling large data-sets
The Fisher Kernel: Jaakkola & Haussler, 1999

Defines a suitable similarity measure for data generated from a probabilistic model $p_{\theta}(x)$ (e.g. GMM)

- Feature vector: $U_{\theta}(x) = \nabla_{\theta} \log p_{\theta}(x)$
  i.e. Fisher Score.

- The Natural Kernel: $k_{\text{nat}}^{M}(x, x') = U_{\theta}(x) M^{-1} U_{\theta}(x')$
  where $M$ is a positive definite matrix.

- The Fisher Kernel: $M = E_{p} \left( U_{\theta}(x) U_{\theta}(x)^{T} \right)$
  i.e. Fisher information matrix.

- In a classification setting with GMMs each class (or Gaussian) can have its own Fisher feature space: $U_{\theta_{c}}(x|c) = \nabla_{\theta_{c}} \log p_{\theta_{c}}(x|c)$. 

18
• Need to “calibrate” SVM scores to combine binary decisions.

• Train a sigmoid to convert SVM margin scores to posteriors. Sigmoid outputs posterior probability of positive class.

\[ p(y = +1|f^*) = \frac{1}{1 + \exp(Af^* + B)} \]
- Reduce a multiclass problem to multiple binary classification problems.
- Find a classifier for each binary problem separately.
- Combine the results (e.g., voting) to predict a label \( c \in \mathcal{C} \).

Examples: one-vs-rest - \( C \) binary classifiers.
  - all-pairs - \( C(C - 1)/2 \) binary classifiers.
  - asymmetric-all-pairs - \( C(C - 1) \) binary classifiers.
# ECOC: Training

## Code Matrix

<table>
<thead>
<tr>
<th>Class</th>
<th>Code Words</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tinky-Winky</strong></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Dipsy</strong></td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Laa-Laa</strong></td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Po</strong></td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Training Set

<table>
<thead>
<tr>
<th></th>
<th>h1</th>
<th>h2</th>
</tr>
</thead>
<tbody>
<tr>
<td>((x_1, T))</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>((x_2, P))</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>((x_3, D))</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>((x_4, T))</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>((x_5, L))</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>((x_6, T))</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>((x_7, D))</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>((x_8, T))</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
ECOC: Classification

Apply the binary classifiers \((h_1, h_2, h_3, h_4, h_5)\) to a new sample. The predictions are:

\[
\begin{array}{cccccc}
+ & + & + & + & - \\
\end{array}
\]

“Decode”:

<table>
<thead>
<tr>
<th></th>
<th>+</th>
<th>+</th>
<th>+</th>
<th>+</th>
<th>-</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>L</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>4</td>
</tr>
<tr>
<td>P</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>3</td>
</tr>
</tbody>
</table>

Predict: **Tinky-Winky**
Hybrid GMM/SVM System

GMM and SVM classifiers with roughly the same accuracy produce uncorrelated errors
The Lincoln Lab Handset database: 52 speakers, telephone bandwidth speech, 4 carbon-button microphones.

Training Data: 2 utterances, each 30s long.

Test Data: 10 utterances from each speaker, each about 10s long, of $52 \times 4 \times 10$ tests.

Training: For each training speaker 51 binary SVMs were trained using the Fisher-feature space corresponding to that speaker’s GMM. Total of $52 \times 51$ binary SVMs trained.

Testing: Output the best-scoring speaker where each speaker’s score is the average of the sigmoid output from that speaker’s 51 (or a subset of size $N - 1$ e.g., $N$-best list from GMM) binary SVMs.
## Text-Independent Speaker Identification System

<table>
<thead>
<tr>
<th>System</th>
<th>Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CB1</td>
</tr>
<tr>
<td>Baseline GMM CB1</td>
<td>6.9</td>
</tr>
<tr>
<td>Hybrid GMM/SVM CB1</td>
<td>5.1</td>
</tr>
<tr>
<td>Rel.red.% CB1</td>
<td>25.7</td>
</tr>
<tr>
<td>Baseline GMM CB3</td>
<td>53.0</td>
</tr>
<tr>
<td>Hybrid GMM/SVM CB3</td>
<td>51.8</td>
</tr>
<tr>
<td>Rel.red.% CB3</td>
<td>2.2</td>
</tr>
</tbody>
</table>

Identification error rates on the CB1- and CB3-trained systems for four types of carbon button microphones.
GMM and SVM classifiers with roughly the same accuracy produce uncorrelated errors
• The Lincoln Lab Handset database: 52 speakers, tele-bandwidth speech, 4 carbon-button microphones.

• Training: As before.

• Testing: Compute Mean ($\mu$) and Standard Deviation ($\sigma$) of GMM scores of each speaker. For frames with score “close” to the mean (e.g., score $\in (\mu - \sigma, \mu + \sigma)$) invoke the SVM decision and “nudge” the score by $\pm \gamma \sigma$ where $\gamma$ is the sigmoidal output from the SVM. Output the best-scoring speaker where each speaker’s score is the average of the perturbed GMM scores.
<table>
<thead>
<tr>
<th>System</th>
<th>Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CB1</td>
</tr>
<tr>
<td>Baseline GMM CB1</td>
<td>5.4</td>
</tr>
<tr>
<td>Enhanced GMM/SVM CB1</td>
<td>3.8</td>
</tr>
<tr>
<td>Rel.red.% CB1</td>
<td>28.6</td>
</tr>
<tr>
<td>Baseline GMM CB3</td>
<td>39.5</td>
</tr>
<tr>
<td>Enhanced GMM/SVM CB3</td>
<td>38.9</td>
</tr>
<tr>
<td>Rel.red.% CB3</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Identification error rates on the CB1- and CB3-trained systems for four types of carbon button microphones using MLLT based Baseline system.
Text-independent Speaker Verification

- **Training Data**: Subset of 1996/1999 Switchboard Evaluation Corpus. 4.5 hours of data for background model. 539 target speakers each with two 1-min utterances of enrollment data.

- **Test Data**: 37620 test (utterance, hypothetical-speaker) pairs. Test duration 15-45s. Ratio of target to impostor trials roughly 1:10.

- **Training**: 1024-Gaussian Universal Background Model (UBM). Target speaker GMMs MAP-adapted from UBM. Upto 1024 (1 per Gaussian) SVMs trained per target using UBM Fisher features.

- **Testing**: 37620 verification trials. DCF = $0.99p(\text{miss}) + 0.1p(\text{false-alarm})$

<table>
<thead>
<tr>
<th></th>
<th>GMM Plain</th>
<th>GMM + MLLT</th>
<th>Enhanced GMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCF $10^{-3}$</td>
<td>52.2</td>
<td>48.4</td>
<td>47.8</td>
</tr>
</tbody>
</table>
Why does the hybrid/enhanced system work?

The error regions of the GMM and the SVM systems are significantly de-correlated, even when their performances are comparable.
Summary

- GMM and SVM classifiers seem to produce uncorrelated errors.
- SVMs work well for multiclass problems with ECOC.
- Fisher Kernel is a reasonable choice for kernel in SVM.
- Hybrid GMM/SVM classifiers work well for Speaker ID.
- Enhanced GMM classifiers with SVM hints work well for Speaker ID and Verification.
- Opens possibilities for variable frame rate processing for speech recognition (e.g., use SVM hints to decide on plosives etc. where SVMs can operate at a higher frame rate) and audio-visual processing...
- Ongoing work to apply this to speech recognition...