Text-Prompted Speaker Recognition with Polynomial Classifiers

Bill Campbell
Motorola Labs
Human Interface Lab
Tempe, AZ, USA

Charles Broun
Motorola Labs
Human Interface Lab
Tempe, AZ, USA
### Motivation

#### Applications
- Many embedded applications in Motorola:
  - Authenticate/personalize device: cell phone, telematics system, pager, software radio, computer
  - Live microphone
- Some server applications:
  - Mya server platform

#### Some Needs
- Small Memory Usage
- Computationally Efficient & Easy to Implement
  - Many devices have simple processors or do not have processors available for ASR/ASV
  - Fixed-point operation is needed in many cases
  - Potential for split system--front end (DSR) in device with server back end
- Robustness
Polynomial Classifiers

• Use polynomials as discriminant functions
  \[ f(x) = p(x)^t w \] where:

  \[ p(x) = \text{vector of monomial basis terms} \]
  \[ = [1 \ x_1 \ x_2 \ x_1^2 \ x_1 x_2 \ x_2^2 \ ... ]^t \]

  \[ w = [w_1 \ w_2 \ w_3 \ ... ]^t \]

• \( p(x) \) is the expansion from input space to “feature space.”
**Simple Sequence Scoring**

- Use simple average of classifier output to obtain score (ICASSP ’99).
- **Advantages:**
  - Low computational complexity
  - Simple implementation
Scoring

• Obtaining a computationally scalable system:

\[
\text{score} = \frac{1}{N} \sum_{i=1}^{N} w^t p(x_i) = w^t \left( \frac{1}{N} \sum_{i=1}^{N} p(x_i) \right) = w^t \bar{p}
\]

The score is a simple inner product after computing the average polynomial expansion.
Training Overview

- **Solve:**
  \[ w^* = \arg \min_w \|Mw - o\|_2 \]

- **Classification model:** \( w \)
- **Matrix of polynomial expansions of the input vectors:** \( M \)
- **Ideal output:** \( o \)

Matrix \( M \):

<table>
<thead>
<tr>
<th>Class Data</th>
<th>Class</th>
<th>Anti - Class 1</th>
<th>Anti - Class N</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p(x)^t )</td>
<td>( p(x)^t )</td>
<td>( 1 )</td>
<td>( 1 )</td>
</tr>
<tr>
<td>( w )</td>
<td>( w )</td>
<td>( 1 )</td>
<td>( 1 )</td>
</tr>
<tr>
<td>( o )</td>
<td>( o )</td>
<td>( 0 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( o )</td>
<td>( o )</td>
<td>( 0 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( o )</td>
<td>( o )</td>
<td>( 0 )</td>
<td>( 0 )</td>
</tr>
</tbody>
</table>
Normal Equations Method

- Basic Idea - use matrix partitioning to divide and conquer.
- Basic Equation:
  \[
  \arg \min_w \left\| \begin{bmatrix} M_{spk} \\ M_{imp} \end{bmatrix} w - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right\|_2
  \]

- Normal Equations:
  \[
  \begin{bmatrix} M_{spk}^t \\ M_{imp}^t \end{bmatrix} \begin{bmatrix} M_{spk} \\ M_{imp} \end{bmatrix} w = \begin{bmatrix} M_{spk}^t \\ M_{imp}^t \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}
  \]
  \[
  \left( R_{spk} + R_{imp} \right) w = M_{spk}^t 1
  \]
  \[
  R_{spk} = \sum_{i=1}^{N_{spk}} p(x_i)p(x_i)^t, \quad R_{imp} = \sum_{i=1}^{N_{imp}} p(y_i)p(y_i)^t
  \]
Training Process for Verification

- Speaker Feature Vectors
- Universal Background $r_{imp}$
- Solve Linear Equation
- Speaker Model

Speaker $r_{spk}$
Remarks on Training

- Background size is fixed.
- Training produces *a posteriori* probabilities so that background *model* normalization is not needed.
  - Bayes rule with *a posteriori* probabilities and two classes involves only the in-class probability.
- Computation is reduced if redundancy is eliminated (ICASSP ’99).
- Training involves only one-pass through the data (closed form instead of iterative solution).
Hybrid HMM/Polynomial Method

Basic Questions

• Can we preserve scoring structure?
• Can we preserve training simplicity?
• What is the probabilistic interpretation of scoring?
• What probabilities should be approximated for training?
HMM/Poly. Scoring

- Use polynomials as frame emission probabilities.
- Concatenate user specific models of subwords corresponding to the spoken utterance.
- To maintain scalability the segmentation must be the same for all user models:
  - Assumption allows us to compute an average polynomial expansion for each state.
  - Method: Use text-independent speech recognition to segment speech.
HMM/Poly. Scoring

Input Speech

**twenty | three | ... | seven**

- **Average Polynomial for Segment**
- **Score with Inner Product**
- **Fuse Scalar Scores**
- **Output Score**

Average Polynomial for Segment

Average Polynomial for Segment

Average Polynomial for Segment

Score with Inner Product

Score with Inner Product

Score with Inner Product
Probabilistic Framework

- Training approximates *a posteriori* probability.
- Using this in scoring (assuming independence of observations) gives:

\[
\text{score} = \sum_{i=1}^{n} \log \left( \frac{p(\omega_j \mid x_i)}{p(\omega_j)} \right)
\]

\[\omega_j \quad \text{Speaker } j\]
\[x_i \quad \text{Feature vector}\]

- Approximate \(\log(a) = a - 1\). Throw away constant -1. We get

\[
\text{score} \approx \sum_{i=1}^{n} \frac{p(\omega_j \mid x_i)}{p(\omega_j)} \approx \frac{1}{p(\omega_j)} \sum_{i=1}^{n} w_j^t p(x_i)
\]
HMM/Polynomial System

- **Which emission probability?**

\[ p(\omega_j \mid q_i, x_i) \]

- **Why?**
  - Other probabilities require more approximation “power”:
    - Example: \[ p(\omega_j, q_i \mid x_i) \]
  - Background normalization not needed since:
    \[ p(\omega_j \mid q_i, x_i) + p(\omega_j \mid q_i, x_i) = 1 \]
  - Training algorithm easily extended:
    - Training using speaker vectors corresponding subword against anti-speaker vectors corresponding to subword.
Experiments Setup

• Tested on YOHO – very close to our major applications.

• YOHO database for speaker recognition.
  – Text-prompted
  – Combination-lock phrases; e.g., “26-81-57.”

• Enrollment - Used all 4 enrollment sessions.

• Verification
  – Used all 40 one phrase tests.
  – Combined 4 phrases from same session for 4 phrase test.

• Features: 12 MFCC, 12 Δ-MFCC, 12 ΔΔ-MFCC
## Verification Results

<table>
<thead>
<tr>
<th>MFCC</th>
<th>$\Delta$-MFCC</th>
<th>$\Delta\Delta$-MFCC</th>
<th>Poly. Degree</th>
<th>Avg. EER 1-phrase %</th>
<th>Avg. EER 4-phrase %</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>-</td>
<td>-</td>
<td>3</td>
<td>0.35</td>
<td>0.02</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>-</td>
<td>2</td>
<td>0.29</td>
<td>0.08</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>-</td>
<td>3</td>
<td>0.07</td>
<td>0.01</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>2</td>
<td>0.24</td>
<td>0.09</td>
</tr>
</tbody>
</table>
### Comparison of Two Technologies

<table>
<thead>
<tr>
<th></th>
<th>HMM (Che/Lin)</th>
<th>New Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>EER - 1 phrase</td>
<td>0.62%</td>
<td>0.29%</td>
</tr>
<tr>
<td>EER - 4 phrase</td>
<td>0.04%</td>
<td>0.08%</td>
</tr>
<tr>
<td>ID Rate - 1 Phrase</td>
<td>0.56%</td>
<td>0.51%</td>
</tr>
<tr>
<td>ID Rate - 4 Phrase</td>
<td>0.14%</td>
<td>0.07%</td>
</tr>
<tr>
<td>Model Parameters</td>
<td>14220</td>
<td>5200</td>
</tr>
</tbody>
</table>
Conclusions

- **Novel architecture:**
  - Computationally efficient
  - Text-prompted
  - No cohort or background normalization

- **Novel training method:**
  - Compact
  - Discriminative

- **Accurate:**
  - High accuracy with less parameter usage than similar HMM implementation.