Gaussian Selection Applied to Text-Independent Speaker Verification
Roland Auckenthaler & John Mason
Speech & Image Research Group
University of Wales Swansea

ODYSSEY 2001

www.swansea.ac.uk
Overview

▼ Computational Efficiencies in GMM SV
▼ Application Environments
▼ Gaussian Selection
▼ Experimental Results on GS
▼ Conclusions
Background

Computational Efficiency

- Time
- Memory

Recognition Accuracy

Application Dependent
Application Environments

▼ Network, Multiple Streams
- Multiple simultaneous requests - variable load
- Computation Time, at peak loads

▼ Wireless (Mobile), Single Stream
- Predictable rates
- Computation Time (Power) and Memory (RAM)
Wireless

Battery Power

Computational Efficiency

Recognition Accuracy

Memory RAM

Network

Computational Efficiency

Memory

Peak Loads

Recognition Accuracy
Two General Approaches:

- **Time Sequence** – frame rate
- **Model Size** – search resolution

Some Previous Work:
- van Vuuren & Hermansky RLA2C 98, ICSLP98
- McLaughlin *et al* Eurospeech-99
- Auckenthaler *et al* ICSLP-99

Conclusions:
- 100 frames/sec much faster than necessary
- ~1024 GMM mixture components is “enough”

NB: Text-{	extit{independent}} and data dependent
The Inevitable Trade-off:

Degradation v’s Efficiency Gains

Error Rate

Degradation

Gain

Frame rate
Model size

FR: 10 100
MS: 512 2048
Acoustic Space Resolution

\( > 90\% \) Computation Time in Scoring

- Reduce Search Space
  - Direct Model size - also reduces memory overheads
  - *Effective* model size - clever searching

- Gaussian Selection
  - Applied in ASR
    - [Bocchieri, ICASSP 93, Gales *et al* IEEE Trans SAP 99]
  - Indexing via low-resolution *hash* table
    - Reduced (sub-optimal) search

- Degradation v’s Search Reduction?
Gaussian Selection in a GMM

- GMM SV System with World Model
- Search via a *Small* Front-End Hash-GMM

```
Hash Model
eg 32 components

World Model

Structure Unchanged

Normed Score

Normed Score

Speaker Model
eg 1024 components

Unchanged
```

9
Hash-GMM Operation

Potential gain comes from *Small* front-end Hash GMM (eg 32 components) followed by limited search of WM

Each Hash component has a shortlist of indices specifying components to be searched in the otherwise conventional World Model
Questions:

- Training of the Hash Model?
- Number/Distribution of Indices?
- Performance Degradation?

The first 2 determine computation saving
Hash-GMM Training

Hash Model

Statistics of Index Pairings

World Model

Training Data (WM)

Hash Model

Shortlists
Hash-GMM Training

▼ 3 Approaches Considered

θ Capped Histogram *(GS)*
θ As above but assign each WM component once only *(GS1)*
θ Train Hash Model on World Model components and assign each to nearest Hash component *(GS2)*

▼ Rationale behind GS1 and GS2 is Coverage of WM

▼ GS2 is the Original Strategy of Bocchieri, ICASSP 93
Experimental Set-up

▼ 1024 component WM
  θ UK telephony DB (M + F)

▼ 16 static + 16 delta spectral coefficients
  θ frame rate 62.5 /sec

▼ Odyssey 2001 evaluation set (males only):
  θ 3 mins training
  θ 15 ~ 46 sec. Test, different handset

▼ Hash model 32
  θ Index shortlist: 16, 32, 64, 128
Hash Model 32, Varying Shortlist:

![Graph showing miss probability vs. false alarm probability for different component combinations.](image)
Covering the World Model:
Comments & Conclusions

▼ Capped Histogram Training:

θ Shortlist size: 128, no discernable loss of accuracy
  efficiency gain ratio ~ 6
θ Shortlist size: 32, small loss of accuracy (1 in 17 EER)
  efficiency gain ratio ~ 16

▼ Efficiencies in Computing Time

θ both application environments (Note shortlist memory can be ROM)

▼ Hash Model Covering the World Model:

θ Seems Unnecessary: 32 Hash + 32 Shortlist similar results
θ ASR Original: Hash derived from mixture means seems inferior

▼ Finally, with no shortlist capping: ~500 index pairs, suggesting a reduction in search by ~50% without loss of accuracy