JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

Speaker Recognition

Najim Dehak

Center for Language and Speech Processing
Johns Hopkins University

CENTER FOR LANGUAGE AND SPEECH PROCESSING

Special Thanks: Paola García, Jesús Villalba, Lukas Burget, Fei Wu,

Roadmap

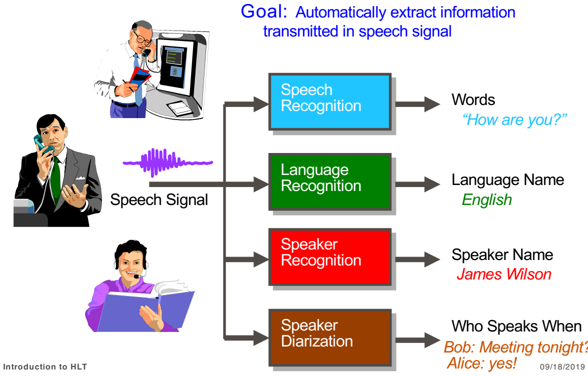
- **Introduction**
  - Terminology, tasks, and framework
- **Low-Dimensional Representation**
  - Sequence of features: GMM
  - Low-dimensional vectors: i-vectors
  - Processing i-vectors: inter-session variability compensation and scoring
  - X-vectors
- **Applications**
  - Speaker verification

Roadmap

- **Introduction**
  - Terminology, tasks, and framework
- **Low-Dimensional Representation**
  - Sequence of features: GMM
  - Low-dimensional vectors: i-vectors
  - Processing i-vectors: compensation and scoring
  - X-vectors
- **Applications**
  - Speaker verification

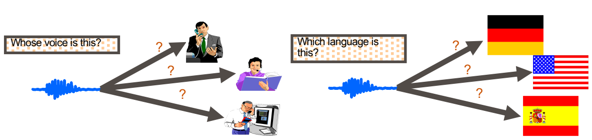
Extracting Information from Speech

Goal: Automatically extract information transmitted in speech signal



Identification

- **Determine whether a test speaker (language) matches one of a set of known speakers (languages)**
- **One-to-many mapping**
- **Often assumed that unknown voice must come from a set of known speakers – referred to as closed-set identification**

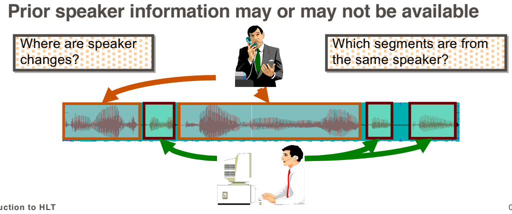


Verification/Authentication

- **Determine whether a test speaker (language) matches a specific speaker (language)**
- **One-to-one mapping**
- **Unknown speech could come from a large set of unknown speakers (languages) – referred to as open-set verification**
- **Adding "unknown class" option to closed-set identification gives open-set identification**

Diarization
Segmentation and Clustering

- **Diarization answers the question: Who speaks when?**
- **Involves:**
  - Determine when a speaker change has occurred in the speech signal (segmentation)
  - Group together speech segments corresponding to the same speaker (clustering)
- **Prior speaker information may or may not be available**



Speech Modalities

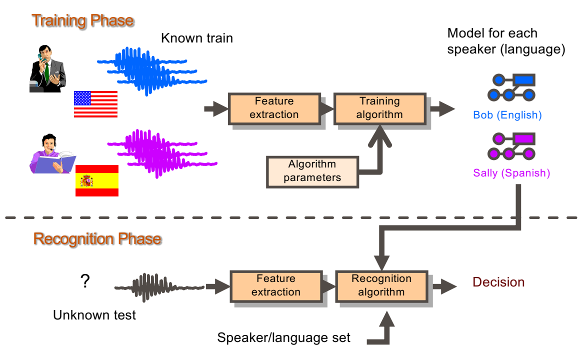
Application dictates different speech modalities:

Text-dependent — Text-independent

- Recognition system knows text spoken by person
- Examples: fixed phrase, prompted phrase
- Used for applications with strong control over user input
- Knowledge of spoken text can improve system performance

- Recognition system does not know text spoken by person
- Examples: User selected phrase, conversational speech
- Used for applications with less control over user input
- More flexible system but also more difficult problem
- Speech recognition can provide knowledge of spoken text

Framework for Speaker/Language Recognition Systems

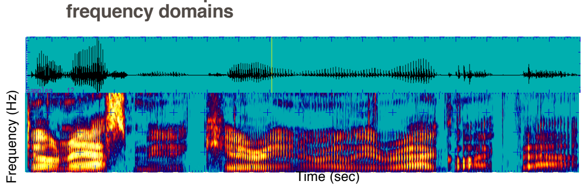


Roadmap

- Introduction
  - Terminology, tasks, and framework
- **Low-Dimensional Representation**
  - Sequence of features: GMM
  - Low-dimensional vectors: i-vectors
  - Processing i-vectors: compensation and scoring
  - X-vectors
- Applications
  - Speaker verification

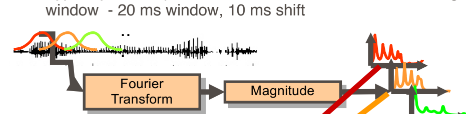
Information in Speech

- **Speech is a time-varying signal conveying multiple layers of information**
  - Words
  - Speaker
  - Language
  - Emotion
- **Information in speech is observed in the time and frequency domains**

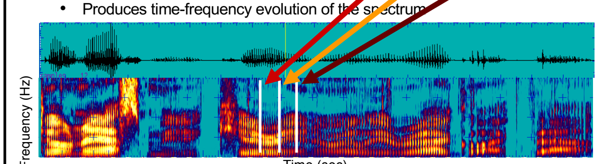


Feature Extraction from Speech

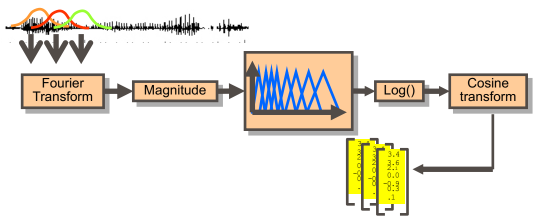
- **A time sequence of features is needed to capture speech information**
  - Typically some spectra based features are extracted using sliding window - 20 ms window, 10 ms shift



- Produces time-frequency evolution of the spectrum



Cepstral Features



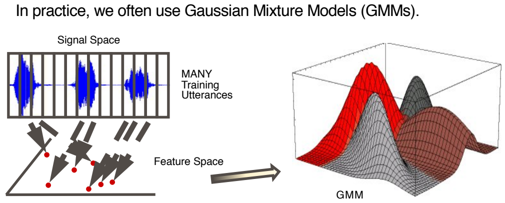
Modeling Sequence of Features
Gaussian Mixture Models

- For most recognition tasks, we need to model the distribution of feature vector sequences

Modeling Sequence of Features
Gaussian Mixture Models

- For most recognition tasks, we need to model the distribution of feature vector sequences

- In practice, we often use Gaussian Mixture Models (GMMs).

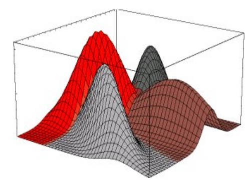


Gaussian Mixture Models

- A GMM is a weighted sum of Gaussian distributions

$$p(\vec{x} \mid \lambda_s) = \sum_{i=1}^{M} p_i b_i(\vec{x})$$

$$\lambda_s = (p_i, \vec{\mu}_i, \Sigma_i)$$



p_i = mixture weight (Gaussian prior proability)
$\vec{\mu}_i$ = mixture mean vector
Σ_i = mixture covariance matrix

$$b_i(\vec{x}) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} \exp(-\tfrac{1}{2}(\vec{x}-\vec{\mu}_i)' \Sigma_i^{-1} (\vec{x}-\vec{\mu}_i))$$

Gaussian Mixture Models
Log Likelihood

- To build a GMM, we need to do two things
  1 – Compute the likelihood of a sequence of features given a GMM
  2 – Estimate the parameters of a GMM given a set of feature vectors

Gaussian Mixture Models
Log Likelihood

- To build a GMM, we need to do two things
  1 – Compute the likelihood of a sequence of features given a GMM
  2 – Estimate the parameters of a GMM given a set of feature vectors
- If we assume independence between feature vectors in a sequence, then we can compute the likelihood as

$$p(\vec{x}_1, \ldots, \vec{x}_N \mid \lambda) = \prod_{n=1}^{N} p(\vec{x}_n \mid \lambda)$$

Gaussian Mixture Models
Log Likelihood

- **Using a GMM involves two things:**
  - 1 – Compute the likelihood of a sequence of features given a GMM
  - 2 – Estimate the parameters of a GMM given a set of feature vectors
- **If we assume independence between feature vectors in a sequence, then we can compute the likelihood as**

$$p(\vec{x}_1,\ldots,\vec{x}_N \mid \lambda) = \prod_{n=1}^{N} p(\vec{x}_n \mid \lambda)$$

- **Usually written as log likelihood**

$$\log p(\vec{x}_1,\ldots,\vec{x}_N \mid \lambda) = \sum_{n=1}^{N} \log p(\vec{x}_n \mid \lambda)$$
$$= \sum_{n=1}^{N} \log\left(\sum_{i=1}^{M} p_i b_i(\vec{x}_n)\right)$$

Gaussian Mixture Models
Parameter Estimation

- **GMM parameters are estimated by maximizing the likelihood given a set of training vectors**

$$\lambda^* = \arg\max_{\lambda} \sum_{n=1}^{N} \log p(\vec{x}_n \mid \lambda)$$

Gaussian Mixture Models
Parameter Estimation

- **GMM parameters are estimated by maximizing the likelihood of on a set of training vectors**

$$\lambda^* = \arg\max_{\lambda} \sum_{n=1}^{N} \log p(\vec{x}_n \mid \lambda)$$

- **Setting the derivatives with respect to model parameters to zero and solving**

$$\Pr(i \mid \vec{x}) = \frac{p_i b_i(\vec{x})}{\sum_{j=1}^{M} p_j b_j(\vec{x})}$$

$$n_i = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_i)$$

$$p_i = \frac{1}{N}\sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)$$

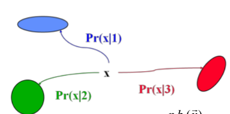
$$\vec{\mu}_i = \frac{1}{n_i}\sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)\vec{x}_n$$

$$\Sigma_i = \frac{1}{n_i}\sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)\vec{x}_i\vec{x}_i{}' - \vec{\mu}_i\vec{\mu}_i{}'$$

Gaussian Mixture Models
Expectation Maximization (EM)

E-Step | M-Step

Probabilistically align vectors to model



$$\Pr(i \mid \vec{x}) = \frac{p_i b_i(\vec{x})}{\sum_{j=1}^{M} p_j b_j(\vec{x})}$$

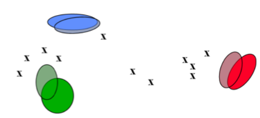
Accumulate sufficient statistics

$$n_i = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)$$

$$E_i(\vec{x}) = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)\vec{x}_n$$

$$E_i(\vec{x}\vec{x}') = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)\vec{x}_n\vec{x}_n{}'$$

Update model parameters



$$p_i = \frac{1}{N} n_i$$

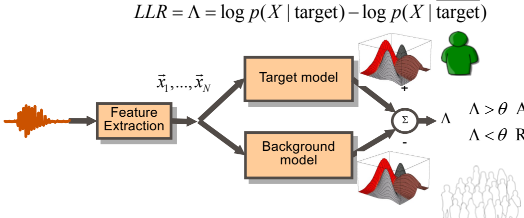
$$\vec{\mu}_i = \frac{1}{n_i} E_i(\vec{x})$$

$$\Sigma_i = \frac{1}{n_i} E_i(\vec{x}\vec{x}') - \vec{\mu}_i\vec{\mu}_i{}'$$

Detection System
GMM-UBM

- Realization of log-likelihood ratio test from signal detection theory

$$LLR = \Lambda = \log p(X \mid \text{target}) - \log p(X \mid \overline{\text{target}})$$



- GMMs used for both target and background model
  - Target model trained using enrollment speech
  - Background model trained using speech from many speakers (often referred to as Universal Background Model – UBM)

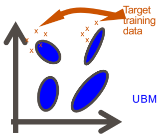
MAP Adaptation

- **Target model is often trained by adapting from background model**
  - Couples models together and helps with limited target training data
- **Maximum A Posteriori (MAP) Adaptation (similar to EM)**
  - Align target training vectors to UBM
  - Accumulate sufficient statistics
  - Update target model parameters with smoothing to UBM parameters
- **Adaptation only updates parameters representing acoustic events seen in target training data**
  - Sparse regions of feature space filled in by UBM parameters
- **Side benefits**
  - Keeps correspondence between target and UBM mixtures (important later)
  - Allows for fast scoring when using many target models (top-M scoring)

Adapted GMMs

- Probabilistically align target training data into UBM mixture states

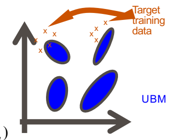
$$\Pr(i \mid \vec{x}) = \frac{p_i b_i(\vec{x})}{\sum_{j=1}^{M} p_j b_j(\vec{x})}$$



Adapted GMMs
Mean-only adaptation

- Probabilistically align target training data into UBM mixture states

$$\Pr(i \mid \vec{x}) = \frac{p_i b_i(\vec{x})}{\sum_{j=1}^{M} p_j b_j(\vec{x})}$$



- Accumulate sufficient statistics from probabilistic alignment
  - Mean-only adaptation empirically found to be better

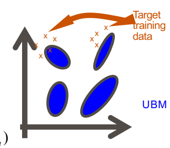
$$n_i = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)$$

$$E_i(\vec{x}) = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)\vec{x}_n$$

Adapted GMMs
Mean-only adaptation

- Probabilistically align target training data into UBM mixture states

$$\Pr(i \mid \vec{x}) = \frac{p_i b_i(\vec{x})}{\sum_{j=1}^{M} p_j b_j(\vec{x})}$$



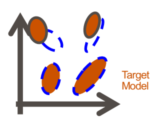
- Accumulate sufficient statistics from probabilistic alignment
  - Mean-only adaptation empirically found to be better

$$n_i = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)$$

$$E_i(\vec{x}) = \sum_{n=1}^{N} \Pr(i \mid \vec{x}_n)\vec{x}_n$$

- Update target model parameters using sufficient statistics and adapt parameter (α)
  - *Relevance factor r controls rate of adaptation*
  - *r → 0, MAP → EM*
  - *r → ∞. No adaptation*

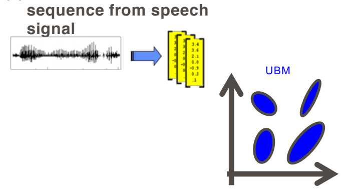
$$\alpha_i = \frac{n_i}{n_i + r}$$



$$\vec{\mu}_i = \alpha_I E_i(\vec{x}) + (1 - \alpha_I)\vec{\mu}_i^{ubm}$$

GMM-UBM Recap

(1) Extract feature vector sequence from speech signal

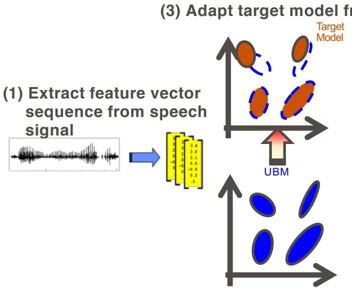


(2) Train UBM with speech from many speakers using EM

GMM-UBM Recap

(3) Adapt target model from UBM

(1) Extract feature vector sequence from speech signal

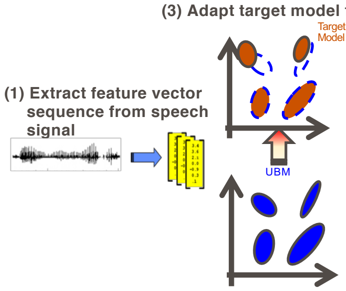


(2) Train UBM with speech from many speakers using EM

GMM-UBM Recap

(3) Adapt target model from UBM

(1) Extract feature vector sequence from speech signal



(4) Compute likelihood ratio of test data

$$LLR(X) = \log p(X \mid \lambda_{target}) - \log p(X \mid \lambda_{ubm})$$

(2) Train UBM with speech from many speakers using EM

Roadmap

- Introduction
  - Terminology, tasks, and framework
- **Low-Dimensional Representation**
  - Sequence of features: GMM
  - Low-dimensional vectors: i-vectors
  - Processing i-vectors: compensation and scoring
  - X-vectors
- Applications
  - Speaker verification

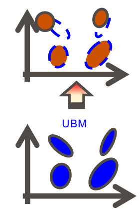
Total variability model (i-vectors)

- The super-vector mean of the GMM of a given recording is written as

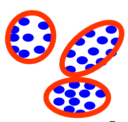
$$\mathbf{M} = \boldsymbol{m} + \mathbf{T}\boldsymbol{w}$$

- w standard Normal random (total factors – intermediate vector or i-vector)
- m : A supervector mean (can be the UBM-GMM)
- **T** : low rank Total variability matrix

$$\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} m_1 \\ m_2 \\ m_1 \\ m_2 \\ m_1 \\ m_2 \end{bmatrix} + \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$



Why call it an i-vector?



GMM components: 2048
Feature dimension: 60

GMM-SV :
60*2048=122880

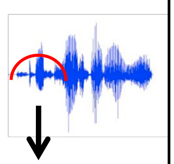
$$\begin{bmatrix} \mu_{11} \\ \mu_{12} \\ \mu_{21} \\ \mu_{22} \\ \mu_{31} \\ \mu_{32} \end{bmatrix}$$

It is definitely not an Apple product

I- for Intermediate representation

I V E C T O R

Actually between 100 to 1000

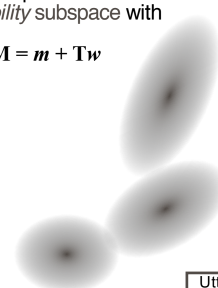


M F C C Feature dimension 60

Visual Interpretation of i-vectors

- To obtain robust estimate of an utterance specific GMM, the mean super-vector is constrained to live in a linear *high variability* subspace with

$$\mathbf{M} = \boldsymbol{m} + \mathbf{T}\boldsymbol{w}$$

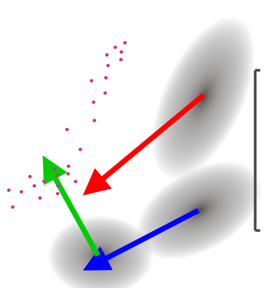


High variability subspace (400 bases)

$$\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} m_1 \\ m_2 \\ m_1 \\ m_2 \\ m_1 \\ m_2 \end{bmatrix} + \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

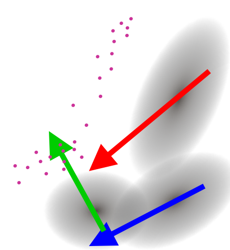
Utterance specific mean super-vector

Visual Interpretation of i-vectors



$$\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} m_1 \\ m_2 \\ m_1 \\ m_2 \\ m_1 \\ m_2 \end{bmatrix} + \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

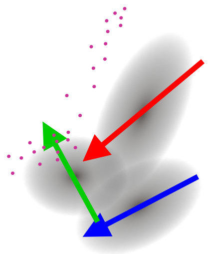
Visual Interpretation of i-vectors



$$\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} m_1 \\ m_2 \\ m_1 \\ m_2 \\ m_1 \\ m_2 \end{bmatrix} + \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} \mathbf{w_1} \\ w_2 \end{bmatrix}$$

Visual Interpretation of i-vectors

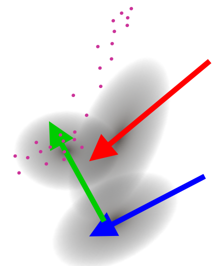
CENTER FOR LANGUAGE AND SPEECH PROCESSING



$$\begin{bmatrix}\mu_1\\ \mu_2\\ \mu_1\\ \mu_2\\ \mu_1\\ \mu_2\end{bmatrix} = \begin{bmatrix}m_1\\ m_2\\ m_1\\ m_2\\ m_1\\ m_2\end{bmatrix} + \begin{bmatrix}t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\end{bmatrix}\begin{bmatrix}w_1\\ w_2\end{bmatrix}$$

Visual Interpretation of i-vectors

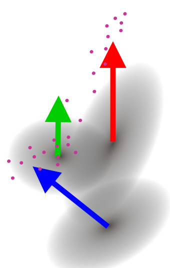
CENTER FOR LANGUAGE AND SPEECH PROCESSING



$$\begin{bmatrix}\mu_1\\ \mu_2\\ \mu_1\\ \mu_2\\ \mu_1\\ \mu_2\end{bmatrix} = \begin{bmatrix}m_1\\ m_2\\ m_1\\ m_2\\ m_1\\ m_2\end{bmatrix} + \begin{bmatrix}t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\end{bmatrix}\begin{bmatrix}w_1\\ w_2\end{bmatrix}$$

Visual Interpretation of i-vectors

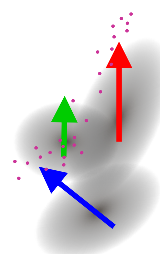
CENTER FOR LANGUAGE AND SPEECH PROCESSING



$$\begin{bmatrix}\mu_1\\ \mu_2\\ \mu_1\\ \mu_2\\ \mu_1\\ \mu_2\end{bmatrix} = \begin{bmatrix}m_1\\ m_2\\ m_1\\ m_2\\ m_1\\ m_2\end{bmatrix} + \begin{bmatrix}t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\end{bmatrix}\begin{bmatrix}w_1\\ w_2\end{bmatrix}$$

Visual Interpretation of i-vectors

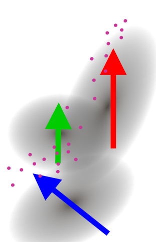
CENTER FOR LANGUAGE AND SPEECH PROCESSING



$$\begin{bmatrix}\mu_1\\ \mu_2\\ \mu_1\\ \mu_2\\ \mu_1\\ \mu_2\end{bmatrix} = \begin{bmatrix}m_1\\ m_2\\ m_1\\ m_2\\ m_1\\ m_2\end{bmatrix} + \begin{bmatrix}t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\end{bmatrix}\begin{bmatrix}w_1\\ w_2\end{bmatrix}$$

Visual Interpretation of i-vectors

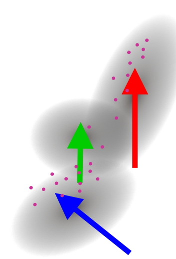
CENTER FOR LANGUAGE AND SPEECH PROCESSING



$$\begin{bmatrix}\mu_1\\ \mu_2\\ \mu_1\\ \mu_2\\ \mu_1\\ \mu_2\end{bmatrix} = \begin{bmatrix}m_1\\ m_2\\ m_1\\ m_2\\ m_1\\ m_2\end{bmatrix} + \begin{bmatrix}t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\end{bmatrix}\begin{bmatrix}w_1\\ w_2\end{bmatrix}$$

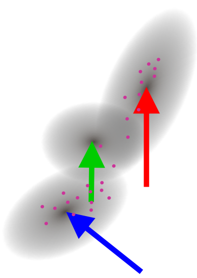
Visual Interpretation of i-vectors

CENTER FOR LANGUAGE AND SPEECH PROCESSING



$$\begin{bmatrix}\mu_1\\ \mu_2\\ \mu_1\\ \mu_2\\ \mu_1\\ \mu_2\end{bmatrix} = \begin{bmatrix}m_1\\ m_2\\ m_1\\ m_2\\ m_1\\ m_2\end{bmatrix} + \begin{bmatrix}t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\\ t_{11} & t_{12}\\ t_{21} & t_{22}\end{bmatrix}\begin{bmatrix}w_1\\ w_2\end{bmatrix}$$

Visual Interpretation of i-vectors



$$\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} m_1 \\ m_2 \\ m_1 \\ m_2 \\ m_1 \\ m_2 \end{bmatrix} + \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} \mathbf{w_1} \\ \mathbf{w_2} \end{bmatrix}$$

Advantages

- **Robustness:**
  - Limiting the adaptation directions of the UBM makes the model more robust to noise, reverberation and other artifacts of the signal
- **Requires less data than GMM-UBM**
  - For GMM-UBM, to adapt all the Gaussians the recording needs to be long enough to contain several frames for all the Gaussians.
  - For i-vectors, we don't need to have data for all the Gaussians.
    - **Use data from a few Gaussians to estimate w**
    - **Use M=m+Tw to get the positions of the unseen Gaussians**
- **Compression:**
  - We summarize a recording of several MB into a small vector.
  - The i-vector is a new feature for other machine learning algorithms

i-vector Calculus

- **In practice, the i-vector is computed using the Bayes Theorem:**
  - We get the posterior distribution for w as

$$P(w|X) = \frac{P(X|w)P(w)}{P(X)} = \frac{\prod_t P(x_t|m+\mathbf{T}w,\Sigma)N(w|0,I)}{P(X)} = \cdots = N(w|E[w],l^{-1})$$

  - The i-vector is the mean $\hat{w} = E[w]$ of the posterior distribution
  - What is the formula for $E[w]$ and l ?

Baum-Welch (Sufficient) Statistics

- **Gaussian responsibilities**

$$\gamma_t(c) = P(c \mid \vec{x}_t, \theta_{\text{UBM}}) = \frac{\pi_c P_c(\vec{x}_t \mid \mu_c, \Sigma_c)}{\sum_{i=1}^{C} \pi_i P_i(\vec{x}_t \mid \mu_i, \Sigma_i)}$$

- **Zeroth Order** $$N_c(u) = \sum_{t=1}^{L} P(c \mid \vec{x}_t, \theta_{\text{UBM}}) = \sum_t \gamma_t(c)$$
- **First Order** $$F_c(u) = \sum_{t=1}^{L} P(c \mid \vec{x}_t, \theta_{\text{UBM}}) \cdot \vec{x}_t = \sum_t \gamma_t(c) \cdot \vec{x}_t$$
- **Centered First order:** $$\tilde{F}_c(u) = \sum_t \gamma_t(c) \cdot (\vec{x}_t - m_c)$$

where $c = 1, \ldots, C$ for each UBM component

Some more notation

$$N(u) = \begin{bmatrix} N_1(u) \cdot I_{F\times F} & 0 & \cdots & 0 \\ 0 & N_2(u) \cdot I_{F\times F} & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & N_C(u) \cdot I_{F\times F} \end{bmatrix}$$

F is the dim of MFCC

$$\tilde{F}(u) = \begin{bmatrix} \tilde{F}_1(u) \\ \tilde{F}_2(u) \\ \vdots \\ \tilde{F}_C(u) \end{bmatrix}$$

The i-vector Calculus

- **Finally the mean of the w Gaussian Posterior is**

$$E[w(u)] = l^{-1}(u) T^t \Sigma^{-1} \tilde{F}(u)$$

and covariance matrix

$$\operatorname{cov}(w(u), w(u)) = l^{-1}(u)$$

where

$$l(u) = I + T^t \Sigma^{-1} N(u) T$$

Kenny, P., Boulianne, G. and P. Dumouchel. Eigenvoice Modeling with Sparse Training Data. IEEE Transactions on Speech and Audio Processing, 13 May (3) 2005 : 345-359.

The EM Algorithm

- **Initialize m and Σ as defined by our UBM covariance matrices**
- **Pick a desired rank R for the Total Variability Matrix T and initialize this CF x R matrix randomly.**
- **E-step:**
  – For each utterance u, calculate the parameters of the posterior distribution of w(u) using the current estimates of m, T, Σ
- **M-step:**
  – Update T solving a set of linear equations in which the w(u)'s play the role of explanatory variables
- **Iterate until parameters / data likelihood converges…**

Kenny, P., Boulianne, G. and P. Dumouchel. Eigenvoice Modeling with Sparse Training Data. IEEE Transactions on Speech and Audio Processing, 13 May (3) 2005 : 345-359.

The M-step

- **In the M-step we maximize the objective function**

$$P(X|\mathbf{T}) \geq Q(\mathbf{T},\mathbf{T}_o) = \sum_u \mathbf{E}[\log P(X_u, w_u|\mathbf{T})|P(w_u|X_u,\mathbf{T}_0)]$$

  – Differentiate and isolate T

$$\frac{\partial Q(\mathbf{T},\mathbf{T}_0)}{\partial T} = 0 \Longrightarrow \mathbf{T}$$

  – Computing T involves solving one linear equation system per Gaussian in the GMM.

Roadmap

- Introduction
  – Terminology, tasks, and framework
- **Low-Dimensional Representation**
  – Sequence of features: GMM
  – Low-dimensional vectors: i-vectors
  – Processing i-vectors: compensation and scoring
  – X-vectors
- Applications
  – Speaker verification

Scoring and channel Compensation

- **Cosine scoring**

$$score = \frac{< w_{\text{enroll}}, w_{\text{test}} >}{\|w_{\text{enroll}}\| \|w_{\text{test}}\|}$$

- **Channel Compensation techniques**
  – Linear Discriminant Analysis
  – Within Class Covariance Normalization [Hatch2006]
  – Nuisance Attribute projection [Campbell 2006]

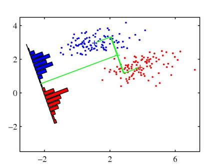
Intersession compensation

- **LDA [Dehak 2009,2011]**

A is matrix of eigenvectors from $S_b.v = \lambda.S_w.v$

$$S_b = \sum_{j=1}^{S}(w_j - \overline{w})(w_j - \overline{w})^t$$

$$S_w = \sum_{s=1}^{S}\frac{1}{n_s}\sum_{i=1}^{n_s}(w_i^s - w_s)(w_i^s - w_s)^t$$

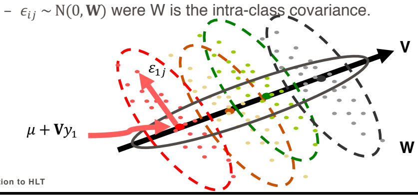


Probabilistic Linear discriminant Analysis (PLDA)

- **Probabilistic version of LDA**
- **i-vector *j* of class *i* is decomposed as a sum of several terms**

$$w_{ij} = \mu + \mathbf{V}y_i + \epsilon_{ij}$$

  – μ is the class-independent mean of all the i-vectors
  – **V** is low rank matrix defining the inter-class variability space
  – $y_i \sim \mathrm{N}(0, \mathrm{I})$ are the coordinates of the speaker in the space defined by **V**
  – $\epsilon_{ij} \sim \mathrm{N}(0, \mathbf{W})$ were W is the intra-class covariance.



PLDA Evaluation

- **Evaluation based on Bayesian model comparison**
  - Likelihood ratio between two hypothesis:
    - **Probability for enrollment and test i-vectors were generated by the same speaker (have the same y)**
    - **Probability for enrollment and test i-vectors were generated by different speakers (have different y)**

$$\mathrm{LLR} = \log \frac{P(w_1,w_2|same)}{P(w_1,w_2|diff)} = \log \frac{\int P(W_1|y)P(W_2|y)P(y)dy}{\int P(W_1|y)P(y)dy \int P(W_2|y)P(y)dy} =$$

$$\log \frac{\int N(w_1|\mu+\mathbf{V}y,W)N(w_2|\mu+\mathbf{V}y,W)N(y|0,I)dy}{\int N(w_1|\mu+\mathbf{V}y,W)N(y|0,I)dy \int N(w_2|\mu+\mathbf{V}y,W)N(y|0,I)\,dy}$$

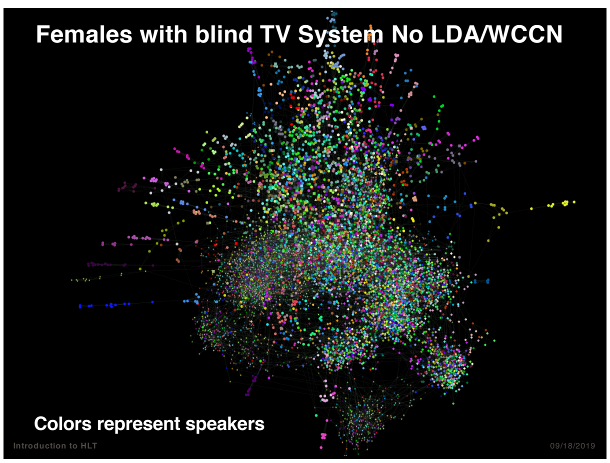
  - In practice, the LLR is a quadratic equation:

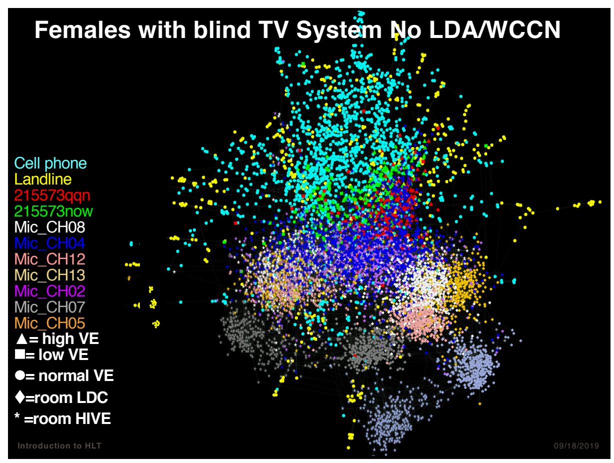
$$LLR = w_1^T A w_2 + \mathrm{w}_1^{\mathrm{T}}\mathrm{B}\mathrm{w}_1 + \mathrm{w}_2^{\mathrm{T}}\mathrm{B}\mathrm{w}_2 + \mathrm{C}^{\mathrm{T}}\mathrm{w}_1 + \mathrm{C}^{\mathrm{T}}\mathrm{w}_2 + \mathrm{D}$$

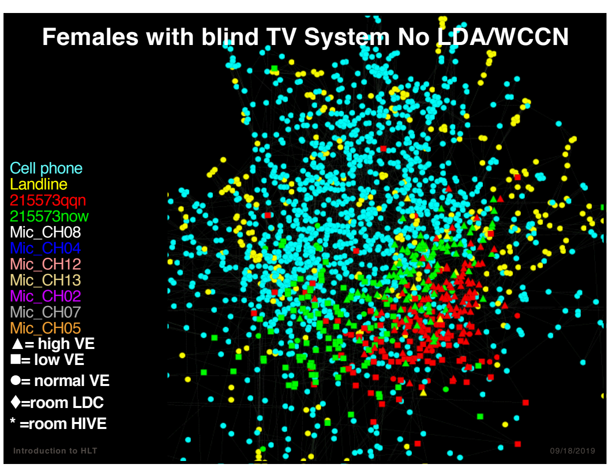
  - μ, V and W are trained using EM algorithm

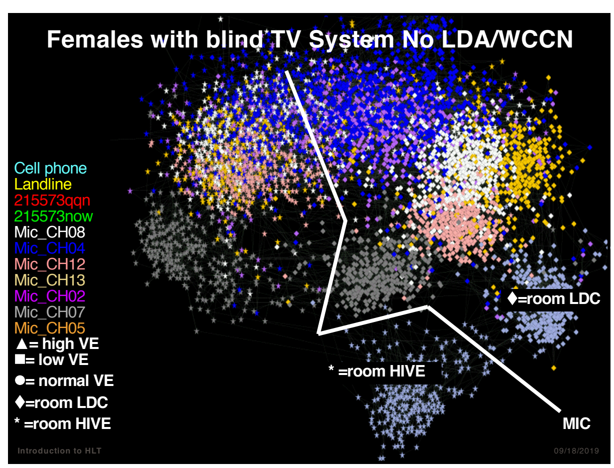
Graph Visualization

- **Work at exploring behavior of speaker matching for large data set mining (Zahi Karam)**
  - Visualization using the Graph Exploration System (GUESS) [Eytan 06]
- **Represent segment as a node with connections (edges) to nearest neighbors (3 NN used)**
  - NN computed using blind TV system (with and without channel normalization)
- **Applied to 5438 utterances from the NIST SRE10 core**
  - Multiple telephone and microphone channels
- **Absolute locations of nodes not important**
- **Relative locations of nodes to one another is important:**
  - The visualization clusters nodes that are highly connected together
- **Colors and shapes of nodes used to highlight interesting phenomena**

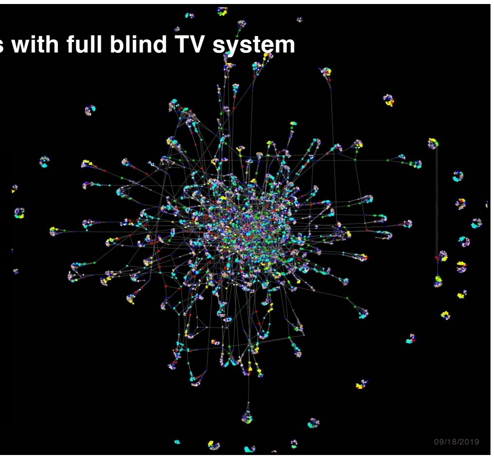






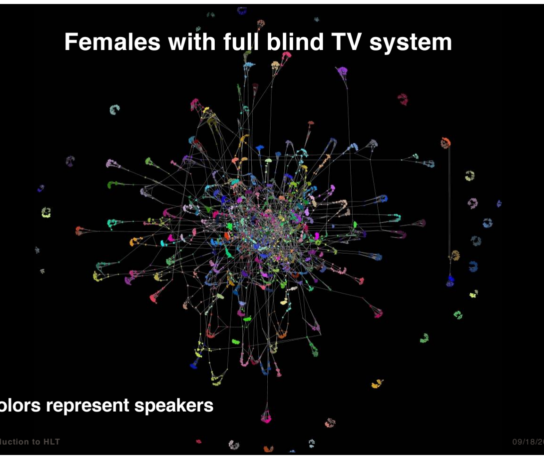


Females with full blind TV system



Cell phone
Landline
215573qqn
215573now
Mic_CH08
Mic_CH04
Mic_CH12
Mic_CH13
Mic_CH02
Mic_CH07
Mic_CH05
▲= high VE
■= low VE
●= normal VE
♦=room LDC
* =room HIVE

Females with full blind TV system



Colors represent speakers

Roadmap

CENTER FOR LANGUAGE AND SPEECH PROCESSING

- Introduction
  - Terminology, tasks, and framework
- Low-Dimensional Representation
  - Sequence of features: GMM
  - Low-dimensional vectors: i-vectors
  - Processing i-vectors: compensation and scoring
  - X-vectors
- Applications
  - Speaker verification

x-Vectors

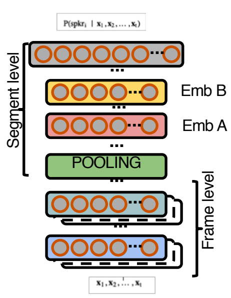
CENTER FOR LANGUAGE AND SPEECH PROCESSING

- Motivation:
  - Can we improve performance by using non-linear models?
  - DNN trained to discriminate between speakers to produce better embeddings.
- Objective:
  - The objective function is cross-entropy $L = -\sum_{i=1}^{M} \log P(y_i = t_i | \mathbf{X}_i)$
  - At the input we have feature sequences of variable length (MFCCs, Mel filter-banks, Bottleneck features)
  - The output of the DNN is the posterior probability for the speaker labels.
  - Requires more training data than i-vectors.
    * **Otherwise it over-fits to training speakers**
    * **Augmenting training data with noise and reverberation improves**

x-Vectors

CENTER FOR LANGUAGE AND SPEECH PROCESSING

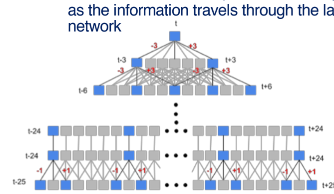
- This DNN has three parts:
  - Encoder: extracts frame level representations
  - Pooling: pooling layer that computes mean and standard deviation.
  - Classification: predicts posterior probabilities for the target speakers
- Once trained:
  - The softmax layer is removed.
  - Embeddings are extracted from the layers after the pooling layer.
  - Typically x-vectors are extracted from the first layer after pooling before applying the non-linear activation function

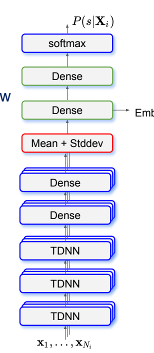


TDNN x-Vector

CENTER FOR LANGUAGE AND SPEECH PROCESSING

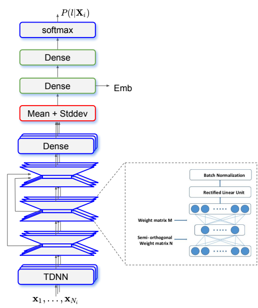
- X-vector inside
  - TDNN encoder
    * TDNN is 1-d dilated convolutional neural network
    * Has the ability to capture features in a wider window as it gets deeper
    * Dilation makes the temporal context to grow faster as the information travels through the layers of the network





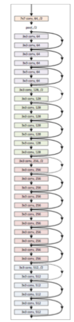
F-TDNN x-Vector

- Factorized TDNN with skip connections
  - Factorizes the weight matrix of each TDNN layer into the product of two low-rank matrices.
    - Reduces network parameters
  - First factor constrained to be semi-orthogonal
    - Matrix rows orthogonal between them
    - Assures that neurons in the bottleneck don't learn redundant information.
  - Skip-connections
    - Between bottleneck representations
    - Representations are concatenated instead of added
    - Allows to make network deeper by alleviating vanishing gradients

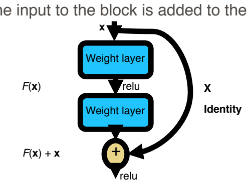


ResNet x-Vector

- **Resnet**



- TDNN encoders are replaced by residual networks (ResNet)
- The MFCCs are replaced by log-Mel filter banks
- ResNet are two-dimensional convolutions (2D-CNN)
- The residual block composed of two 2D convolutions separated by a ReLU
- The input to the block is added to the output



x-Vector Temporal Pooling

- **Pooling methods**
  - **Mean+Standard Deviation:**
    - Standard method computes mean and stddev of frame level representations over time
  - **Learnable dictionary encoder (LDE)**
    - Frame level representations are modeled as a GMM (Similar to i-vectors)
    - The probability that frame t belongs to Gaussian component c is

$$w_{t,c} = \frac{\exp(-s_c \|\mathbf{x}_t - \boldsymbol{\mu}_c\|^2 + b_c)}{\sum_{c=1}^{C} \exp(-s_c \|\mathbf{x}_t - \boldsymbol{\mu}_c\|^2 + b_c)}$$

    - Compute one embedding per component by averaging the frames of that component
    - Concatenate component embeddings to form a super-vector

$$\mathbf{e}_c = \frac{\sum_{t=1}^{T} w_{t,c}(\mathbf{x}_t - \boldsymbol{\mu}_c)}{\sum_{t=1}^{T} w_{t,c}} \qquad c = 1, \ldots, C$$

$$\mathbf{e} = (\mathbf{e}_1^{\mathrm{T}}, \ldots, \mathbf{e}_C^{\mathrm{T}})^{\mathrm{T}}$$

  - **Multi-head Attention**
    - Similar to LDE but weights are normalized to sum up to one over time.
    - Attends to the most important frames in the sequence for cluster c

$$w_{t,c} = \frac{\exp(-s_c \|\mathbf{x}_t - \boldsymbol{\mu}_c\|)}{\sum_{t=1}^{T} \exp(-s_c \|\mathbf{x}_t - \boldsymbol{\mu}_c\|)}$$

X-vector

- Backend:
  - LDA,
  - Centering, whitening
  - length normalization
  - PLDA scoring
- Same back-end as the one used for i-vectors.

Discussion

- **Low dimensional representation simplifies life**
- **i/x-Vector transforms a sequence of features into a unique vector**
- **Easy way to compare between sequences of features with different duration**
- **Classical pattern recognition approaches like LDA, PLDA or SVM can be used to compare i/x-vectors**
- **X-vectors are now the state-of-the-art.**

Roadmap

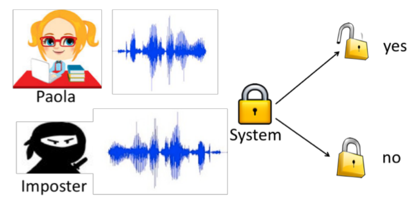
- Introduction
  - Terminology, tasks, and framework
- Low-Dimensional Representation
  - Sequence of features: GMM
  - Low-dimensional vectors: i-vectors
  - Processing i-vectors: compensation and scoring
  - X-vectors
- **Applications**
  - Speaker verification

JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

Speaker Verification

CENTER FOR LANGUAGE AND SPEECH PROCESSING

Speaker Verification Problem



Speaker Verification

Speaker Verification: Accepts or rejects a user based on his speech signal.

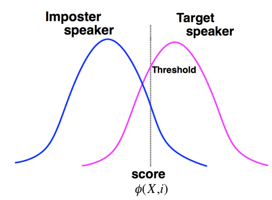
- Input:
  - Speech signal X 
  - Claimed identity i Paola
- Output:

$$d = \begin{cases} \text{accept} & \phi(X,i) > \tau_i; \\ \text{reject} & \text{otherwise} \end{cases}$$

$\phi(X,i)$ is a confidence measure

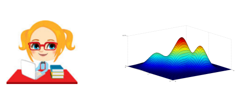
Score Distribution

- Binary classifier with the following confidence measures (*scores*).
- The rightmost Gaussian belongs to the *target speaker*.
- The leftmost Gaussian belongs to the *imposter speaker*.
- Key point: a decision threshold.

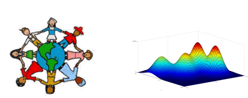


Speaker Verification: What is needed?

- Each accredited speaker has its own model, known as target model, λ_i , prototype of his/her speech.



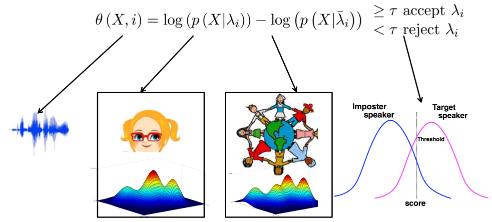
- And an imposter model $\overline{\lambda_i}$ is the impostor's prototype. When all the imposters share the same model (they are "tied"), called: UBM Universal Background Model.



Log-Likelihood Ratio

- The likelihood ratio provides a tool to perform a statistical decision (score function in log domain) :

$$\theta(X,i) = \log\left(p\left(X|\lambda_i\right)\right) - \log\left(p\left(X|\bar{\lambda}_i\right)\right) \begin{array}{l} \geq \tau \text{ accept } \lambda_i \\ < \tau \text{ reject } \lambda_i \end{array}$$



Hypothesis Testing

Hypotheses Testing is a suitable framework for detection problems:

- $H0$, the null hypothesis, accepts the identity of the speaker as *legitimate*.

- $H1$, the alternative hypothesis, rejects the user (*imposter*).

What if something goes wrong in the system?

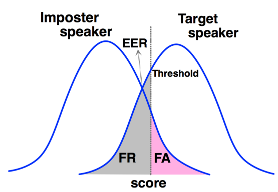
Types of Errors

For a classifier, there are two sources of statistical errors:

- If *H0* is rejected when *H0* is actually from the speaker (reject a legitimate user), false negative, miss or false rejected.

- If it fails to reject *H1*, when *H1* is false (accepts an impostor), false positive (FP), false alarm or false accepted.

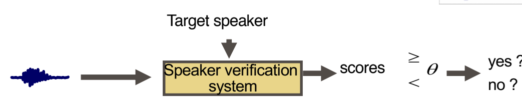
Types of Errors



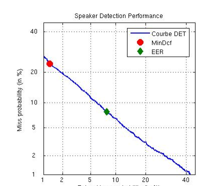
The main goal for speaker verification must be to minimize those errors.

The tradeoff between the errors depend on the application.

Speaker verification system performances



- **Detcurve**



- False acceptance and rejection Rates

$$R_{FA} = \frac{\text{Number of False Acceptance}}{\text{Number of impostors accesses}}$$

$$R_{FR} = \frac{\text{Number of False Rejection}}{\text{Number of target accesses}}$$

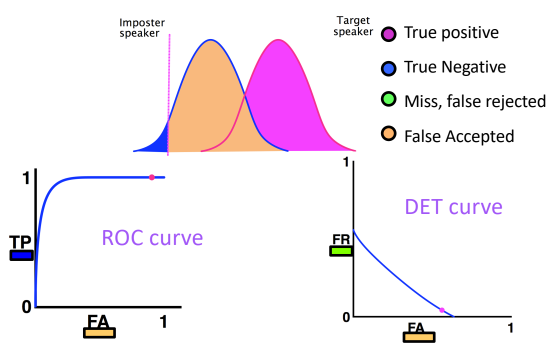
- **EER**

$$R_{FA} = R_{FR}$$

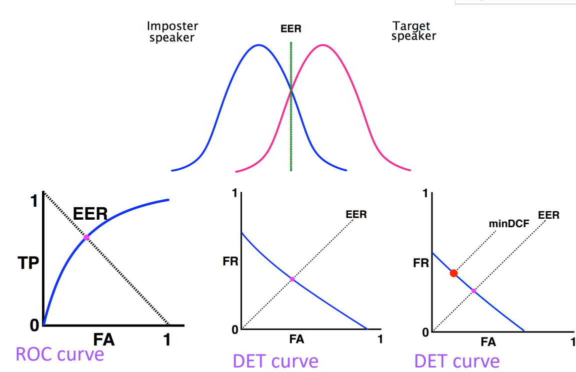
- **MinDCF**

$$DCF = C_{FR}.P_{target}.R_{FR} + C_{FA}.P_{imposteur}.R_{FA}$$

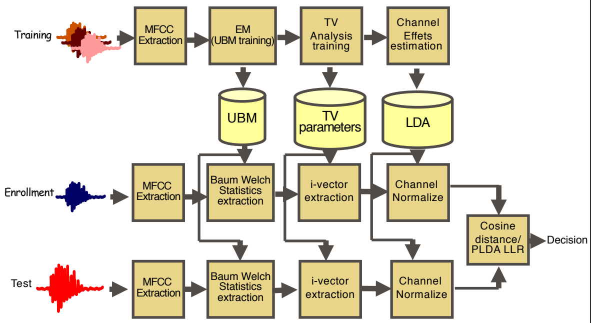
ROC vs DET curves



Metrics



i-vector System



GMM i-vector vs DNN i-vector

- NIST SRE10, five conditions, females
- 2048 component UBM, 600 dimensional i-vector
- DNN trained on 250 hours of Fisher

	Condition 1 (int-int same mic.)			Condition 2 (int-int diff. mic.)		
	minDCF10	minDCF08	EER (%)	minDCF10	minDCF08	EER (%)
GMM-UBM	0.183	0.051	1.30	0.311	0.088	1.94
DNN-UBM	0.142	0.032	0.77	0.205	0.053	1.32

	Condition 3 (int-tel)			Condition 4 (int-mic)		
	minDCF10	minDCF08	EER (%)	minDCF10	minDCF08	EER (%)
GMM-UBM	0.316	0.091	2.07	0.223	0.050	1.00
DNN-UBM	0.204	0.049	1.18	0.130	0.024	0.53

	Condition 5 (tel-tel)		
	minDCF10	minDCF08	EER (%)
GMM-UBM	0.390	0.110	2.21
DNN-UBM	0.209	0.056	1.21

X-vectors

- Some results…

Systems	SRE18 DEV CMN2			SRE18 EVAL CMN2		
	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp
GMM-i-vector	10.37	0.664	0.685	11.85	0.723	0.725
BNF-i-vector	10.51	0.639	0.657	11.69	0.71	0.712
TDNN(8.5M)-sre16	7.2	0.505	0.51	7.93	0.515	0.518
TDNN(8.5M)	5.76	0.384	0.392	6.68	0.446	0.447
E-TDNN(10M)	5.88	0.392	0.398	5.97	0.409	0.41
F-TDNN(11M)	**4.96**	**0.326**	**0.33**	5.3	0.37	0.371
F-TDNN(17M)	5.1	0.355	0.372	**4.95**	**0.346**	**0.349**
ResNet(8M)-MHAtt-SPLDA	5.46	**0.326**	**0.34**	5.64	0.392	0.395
ResNet(8M)-MHAtt-DPLDA	5.64	**0.319**	**0.337**	6.81	0.499	0.524

X-vectors

System	SITW EVAL CORE			SITW EVAL CORE-MULTI			SRE18 DEV VAST			SRE18 EVAL VAST		
	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp
16 kHz systems												
BNF-i-vector	5.77	0.257	0.262	6.02	0.26	0.26	11.52	**0.185**	**0.222**	17.46	0.508	0.571
TDNN(8.5M)	3.4	0.185	0.188	3.86	0.191	0.191	**3.7**	0.337	0.424	12.06	0.468	0.578
E-TDNN(10M)	2.74	0.162	0.165	3.2	0.171	0.172	**3.7**	0.305	0.305	13.02	0.442	0.527
F-TDNN(9M)	2.39	0.144	0.15	2.79	0.153	0.153	4.53	0.309	0.383	11.75	0.412	0.508
F-TDNN(10M)	2.37	0.135	0.138	2.86	0.145	0.146	**3.7**	0.337	0.42	**10.79**	0.403	0.503
F-TDNN(11M)	2.05	0.137	0.14	2.57	0.145	0.147	**3.7**	0.305	0.387	11.11	0.409	0.487
F-TDNN(17M)	**1.89**	**0.124**	**0.126**	**2.33**	**0.135**	**0.137**	7	0.37	0.498	12.06	**0.388**	**0.474**
ResNet(8M)	3.01	0.187	0.191	3.47	0.198	0.198	**3.7**	0.412	0.498	11.43	0.464	0.554
8 kHz systems												
GMM-i-vector	8.22	0.384	0.393	8.67	0.386	0.387	18.52	0.486	0.568	20.32	0.543	0.75
BNF-i-vector	7.8	0.353	0.365	8.42	0.352	0.354	14.81	0.412	0.568	17.9	0.533	0.638
TDNN(8.5M)-sre16	5.21	0.278	0.284	5.6	0.287	0.287	11.11	0.3	0.691	13.33	0.475	0.636
TDNN(8.5M)	3.58	0.197	0.202	3.93	0.206	0.207	7.41	0.296	0.535	12.93	0.431	0.596
E-TDNN(10M)	2.9	0.172	0.175	3.29	0.183	0.183	7.41	0.337	0.461	12.6	0.41	0.561
F-TDNN(11M)	2.84	0.158	0.163	3.18	0.165	0.166	7.41	**0.222**	0.461	12.06	0.385	0.52
F-TDNN(17M)	**2.46**	**0.148**	**0.151**	**2.83**	**0.155**	**0.156**	**4.53**	0.259	**0.383**	**11.75**	**0.377**	**0.514**