# Speaker Adaptation

Steve Renals

Automatic Speech Recognition – ASR Lecture 11
22 March 2018

# Speaker independent / dependent / adaptive

- Speaker independent (SI) systems have long been the focus for research in transcription, dialogue systems, etc.
- Speaker dependent (SD) systems can result in word error rates 2–3 times lower than SI systems (given the same amount of training data)
- A Speaker adaptive (SA) system... we would like
  - Error rates similar to SD systems
  - Building on an SI system
  - Requiring only a small fraction of the speaker-specific training data used by an SD system

# Speaker-specific variation

- Acoustic model
  - Speaking styles
  - Accents
  - Speech production anatomy (eg length of the vocal tract)

  Also non-speaker variation, such as channel conditions (telephone, reverberant room, close talking mic) and application domain

  Speaker adaptation of acoustic models aims to reduce the mismatch between test data and the models

# Speaker-specific variation

- Acoustic model
  - Speaking styles
  - Accents
  - Speech production anatomy (eg length of the vocal tract)

  Also non-speaker variation, such as channel conditions (telephone, reverberant room, close talking mic) and application domain

  Speaker adaptation of acoustic models aims to reduce the mismatch between test data and the models
- Pronunciation model: speaker-specific, consistent change in pronunciation
- Language model: user-specific documents (exploited in personal dictation systems)

# Modes of adaptation

- Supervised or unsupervised
  - **Supervised**: the word level transcription of the adaptation data is known
  - **Unsupervised**: no transcription provided
- Static or dynamic
  - **Static**: Adaptation data presented to the system in a block before the final system is estimated (eg enrolment in a dictation system)
  - **Dynamic**: Adaptation data incrementally available, models must be adapted before all adaptation data is available (eg spoken dialogue system)
- Desirable properties for speaker adaptation
  - **Compact:** relatively few speaker-dependent parameters
  - **Unsupervised:** does not require labelled adaptation data
  - **Efficient:** low computational requirements
  - **Flexible:** applicable to different model variants

# Approaches to adaptation

- Model based: Adapt the parameters of the acoustic models to better match the observed data
  - Maximum a posteriori (MAP) adaptation of HMM/GMM parameters
  - Maximum likelihood linear regression (MLLR) of GMM parameters
  - Learning Hidden Unit Contributions (LHUC) for neural networks
- Speaker normalization: Normalize the acoustic data to reduce mismatch with the acoustic models
  - Vocal Tract Length Normalization (VTLN)
  - Constrained MLLR (cMLLR) — model-based normalisation
- Speaker space: Estimate multiple sets of acoustic models, characterizing new speakers in terms of these model sets
  - Cluster-adaptive training
  - Eigenvoices
  - Speaker codes

# Approaches to adaptation

- Model based: Adapt the parameters of the acoustic models to better match the observed data
  - **Maximum a posteriori (MAP) adaptation** of HMM/GMM parameters
  - **Maximum likelihood linear regression (MLLR)** of GMM parameters
  - **Learning Hidden Unit Contributions (LHUC)** for neural networks
- Speaker normalization: Normalize the acoustic data to reduce mismatch with the acoustic models
  - Vocal Tract Length Normalization (VTLN)
  - **Constrained MLLR (cMLLR)** — model-based normalisation
- Speaker space: Estimate multiple sets of acoustic models, characterizing new speakers in terms of these model sets
  - Cluster-adaptive training
  - Eigenvoices
  - **Speaker codes**

# Model-based adaptation: MAP training

- Basic idea MAP training balances the parameters estimated on the SI data with estimates from the new data
- Consider the mean of the $m$th Gaussian in the $j$th state, $\boldsymbol{\mu}_{mj}$
  - ML estimation of SI model:

$$\boldsymbol{\mu}_{mj} = \frac{\sum_n \gamma_{jm}(n)\mathbf{x}_n}{\sum_n \gamma_{jm}(n)}$$

where $\gamma_{jm}(n)$ is the component occupation probability

# Model-based adaptation: MAP training

- Basic idea MAP training balances the parameters estimated on the SI data with estimates from the new data
- Consider the mean of the $m$th Gaussian in the $j$th state, $\boldsymbol{\mu}_{mj}$
  - ML estimation of SI model:

$$\boldsymbol{\mu}_{mj} = \frac{\sum_n \gamma_{jm}(n)\mathbf{x}_n}{\sum_n \gamma_{jm}(n)}$$

    where $\gamma_{jm}(n)$ is the component occupation probability
  - **MAP estimate** for the adapted model:

$$\hat{\boldsymbol{\mu}} = \frac{\tau\boldsymbol{\mu}_0 + \sum_n \gamma(n)\mathbf{x}_n}{\tau + \sum_n \gamma(n)}$$

    - $\tau$ controls the balance between SI estimate and the adaptation date Typically $0 \leq \tau \leq 20$
    - $\mathbf{x}_n$ is the adaptation vector at time $n$
    - $\gamma(n)$ the probability of this Gaussian at this time

# Model-based adaptation: MAP training

- Basic idea MAP training balances the parameters estimated on the SI data with estimates from the new data
- Consider the mean of the $m$th Gaussian in the $j$th state, $\boldsymbol{\mu}_{mj}$
  - ML estimation of SI model:

$$\boldsymbol{\mu}_{mj} = \frac{\sum_n \gamma_{jm}(n)\mathbf{x}_n}{\sum_n \gamma_{jm}(n)}$$

    where $\gamma_{jm}(n)$ is the component occupation probability
  - **MAP estimate** for the adapted model:

$$\hat{\boldsymbol{\mu}} = \frac{\tau\boldsymbol{\mu}_0 + \sum_n \gamma(n)\mathbf{x}_n}{\tau + \sum_n \gamma(n)}$$

    - $\tau$ controls the balance between SI estimate and the adaptation date Typically $0 \leq \tau \leq 20$
    - $\mathbf{x}_n$ is the adaptation vector at time $n$
    - $\gamma(n)$ the probability of this Gaussian at this time

# Model-based adaptation: MAP training

- Basic idea MAP training balances the parameters estimated on the SI data with estimates from the new data
- Consider the mean of the $m$th Gaussian in the $j$th state, $\boldsymbol{\mu}_{mj}$
  - ML estimation of SI model:

$$\boldsymbol{\mu}_{mj} = \frac{\sum_n \gamma_{jm}(n)\mathbf{x}_n}{\sum_n \gamma_{jm}(n)}$$

    where $\gamma_{jm}(n)$ is the component occupation probability
  - **MAP estimate** for the adapted model:

$$\hat{\boldsymbol{\mu}} = \frac{\tau\boldsymbol{\mu}_0 + \sum_n \gamma(n)\mathbf{x}_n}{\tau + \sum_n \gamma(n)}$$

    - $\tau$ controls the balance between SI estimate and the adaptation date Typically $0 \leq \tau \leq 20$
    - $\mathbf{x}_n$ is the adaptation vector at time $n$
    - $\gamma(n)$ the probability of this Gaussian at this time
  - As the amount of training data increases, so the MAP estimate converges to the ML estimate

# The MLLR family

- Basic idea Rather than directly adapting the model parameters, to apply to the Gaussian means and covariances
- MAP training only adapts parameters belonging to observed components – with many Gaussians and a small amount of adaptation data, most Gaussians are not adapted
- Solution: share adaptation parameters across Gaussians – each adaptation data point can then affect many of (or even all) the Gaussians in the system
- Since there are relatively few adaptation parameters, estimation is robust
- **Maximum Likelihood Linear Regression (MLLR)** – use a linear transform to share adaptation parameters across Gaussians

# MLLR: Maximum Likelihood Linear Regression

- MLLR adapts the means of the Gaussians by applying an affine (linear) transform of mean parameters

$$\hat{\mu} = \mathbf{A}\mu + \mathbf{b}$$

  If the observation vectors are $d$-dimension, then $\boldsymbol{A}$ is a $d \times d$ matrix and $\mathbf{b}$ is $d$-dimension vector
- If we define $\mathbf{W} = [\mathbf{bA}]$ and $\eta = [1\mu^T]^T$, then we can write:

$$\hat{\mu} = \mathbf{W}\eta$$

- In MLLR, $\mathbf{W}$ is estimated so as to maximize the likelihood of the adaptation data
- A single transform $\mathbf{W}$ can be shared across a set of Gaussian components (even all of them!)

# How many transforms?

- A set of Gaussian components that share a transform is called a *regression class*
- In practice the number of regression is often small: one per context-independent phone class, one per broad class, two (speech/non-speech), or just a single transform for all Gaussians
- The number of regression classes may also be obtained automatically by constructing a *regression class tree*
  - Each node in the tree represents a regression class sharing a transform
  - For an adaptation set, work down the tree until arriving at the most specific set of nodes for which there is sufficient data
  - Regression class tree constructed in a similar way to state clustering tree

# Estimating the transforms

- The linear transformation matrix $W$ is obtained by finding its setting which optimizes the log likelihood
- Mean adaptation: Log likelihood

$$L = \sum_r \sum_n \gamma_r(n) \log \left( K_r \exp \left( -\frac{1}{2} (\mathbf{x}_n - \mathbf{W}\boldsymbol{\eta}_r)^T \boldsymbol{\Sigma}_r^{-1} (\mathbf{x}_n - \mathbf{W}\boldsymbol{\eta}_r) \right) \right)$$

  where $r$ ranges over the components belonging to the regression class
- Differentiating $L$ and setting to 0 results in an equation for $\mathbf{W}$: there is no closed form solution if $\boldsymbol{\Sigma}$ is full covariance; can be solved if $\boldsymbol{\Sigma}$ is diagonal (but requires a matrix inversion)
- Variance adaptation is also possible
- See Gales and Woodland (1996), Gales (1998) for details

# MLLR in practice

- Mean-only MLLR results in 10–15% relative reduction in WER
- Few regression classes and well-estimated transforms work best in practice
- Robust adaptation available with about 1 minute of speech; performance similar to SD models available with 30 minutes of adaptation data
- Such linear transforms can account for any systematic (linear) variation from the speaker independent models, for example those caused by channel effects.

# Constrained MLLR (cMLLR)

- Basic idea use the same linear transform for both mean and covariance

$$\hat{\mu} = \mathbf{A}'\mu - \mathbf{b}'$$
$$\hat{\mathbf{\Sigma}} = \mathbf{A}'\mathbf{\Sigma}\mathbf{A}'^{T}$$

- No closed form solution but can be solved iteratively
- Log likelihood for cMLLR

$$L = \mathcal{N}(\mathbf{A}\mathbf{x}_n + \mathbf{b}; \mu, \mathbf{\Sigma}) + \log(|\mathbf{A}|) \qquad \mathbf{A}' = \mathbf{A}^{-1}; \mathbf{b}' = \mathbf{A}\mathbf{b}$$

**Equivalent to applying the linear transform to the data!**
Also called *fMLLR (feature space MLLR)*

- Iterative solution amenable to online/dynamic adaptation, by using just one iteration for each increment
- Similar improvement in accuracy to standard MLLR

# Speaker-adaptive training (SAT)

- Basic idea Rather than SI seed (canonical) models, construct models designed for adaptation
- Estimate parameters of canonical models by training MLLR mean transforms for each training speaker
- Train using the MLLR transform for each speaker; interleave Gaussian parameter estimation and MLLR transform estimation
- SAT results in much higher training likelihoods, and improved recognition results
- But: increased training complexity and storage requirements
- SAT using cMLLR, corresponds to a type of speaker normalization at training time

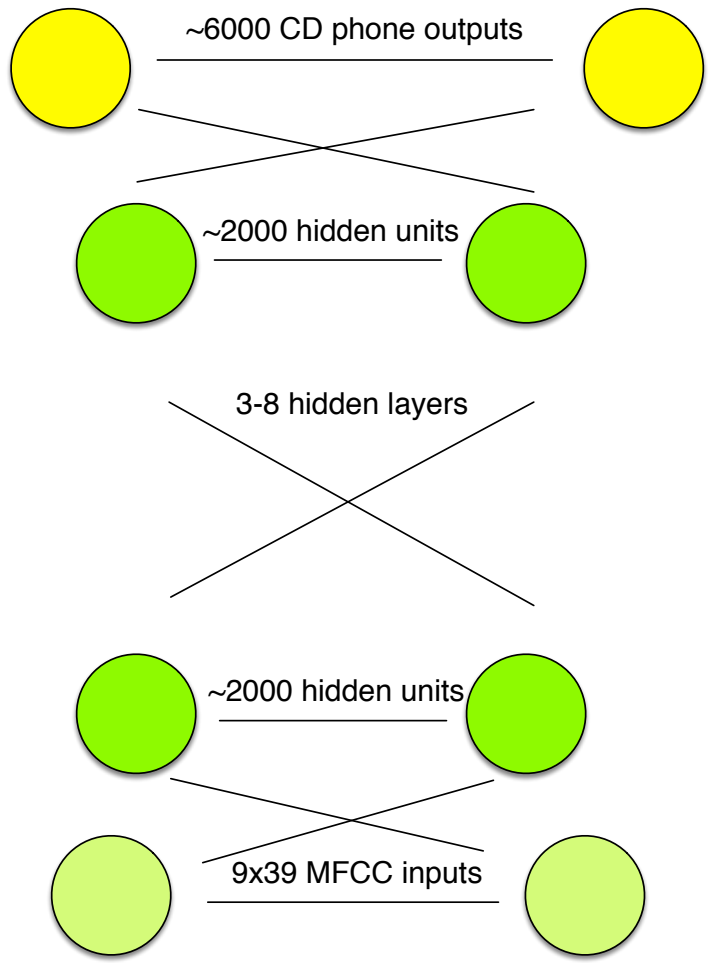
# Speaker adaptation in hybrid HMM/NN systems: CMLLR feature transformation

- Basic idea: If HMM/GMM system is used to estimate a single constrained MLLR adaptation transform, this can be viewed as a feature space transform
- Use the HMM/GMM system with the same tied state space to estimate a single CMLLR transform for a given speaker, and use this to transform the input speech to the DNN for the target speaker
- Can operate unsupervised (since the GMM system estimates the transform)
- Limited to a single transform (regression class)

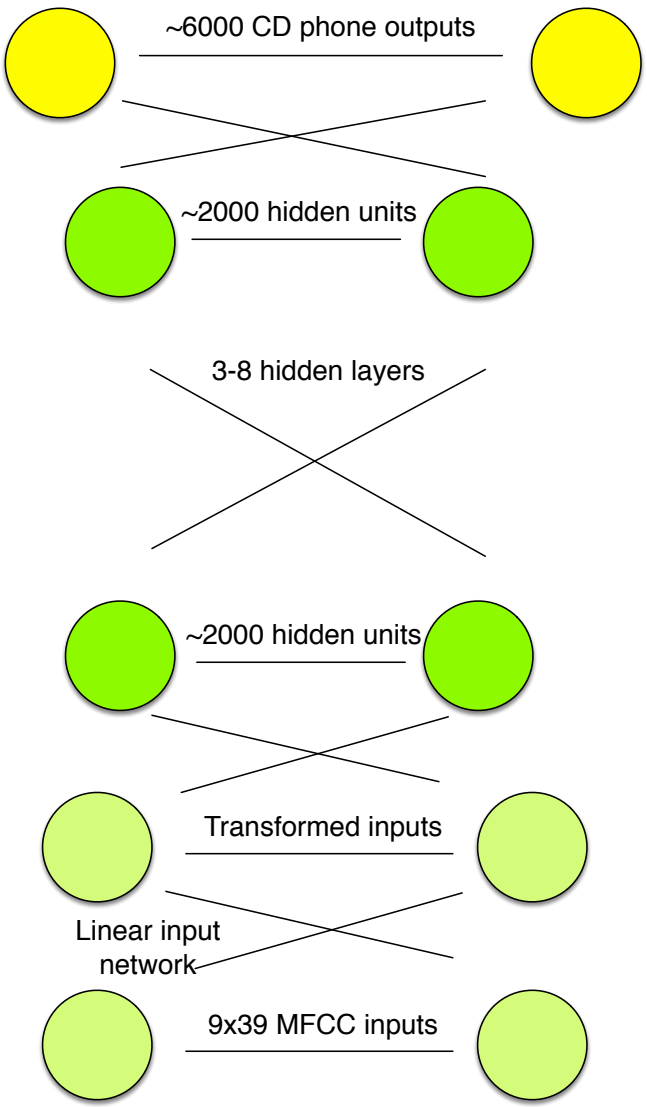
# Speaker adaptation in hybrid HMM/NN systems: LIN – Linear Input Network

- Basic idea: single linear input layer trained to map input speaker-dependent speech to speaker-independent network
- Training: linear input network (LIN) can either be fixed as the identity or (adaptive training) be trained along with the other parameters
- Testing: freeze the main (speaker-independent) network and propagate gradients for speech from the target speaker to the LIN, which is updated — linear transform learned for each speaker
- Requires supervised training data

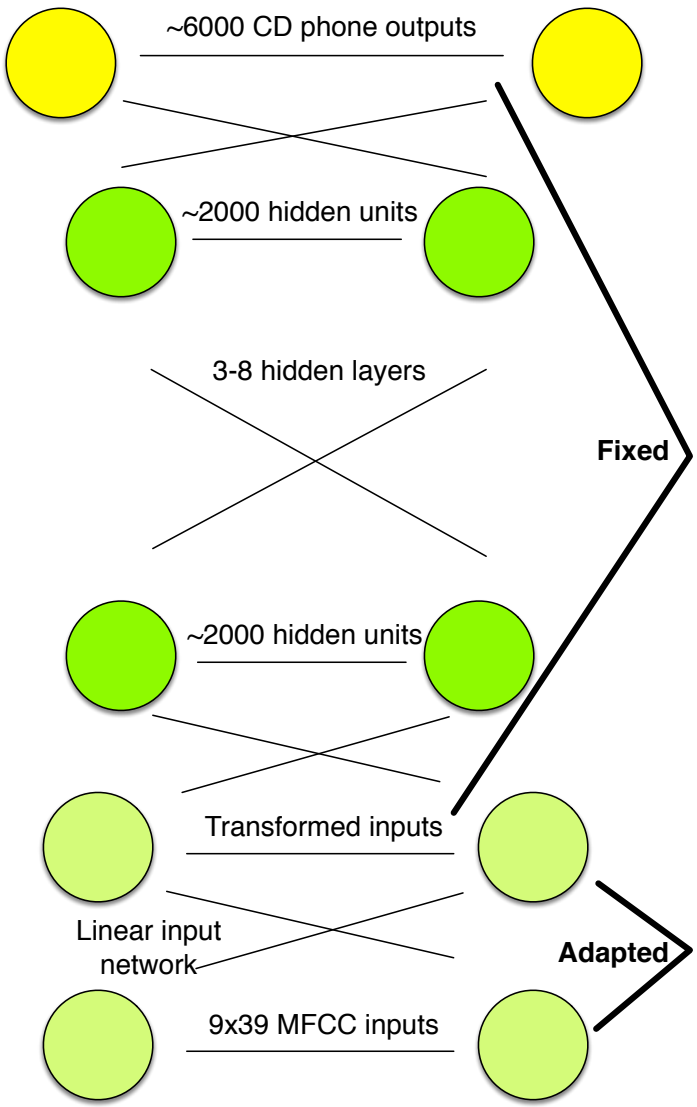
# LIN


~6000 CD phone outputs
~2000 hidden units
3-8 hidden layers
~2000 hidden units
9x39 MFCC inputs

# LIN

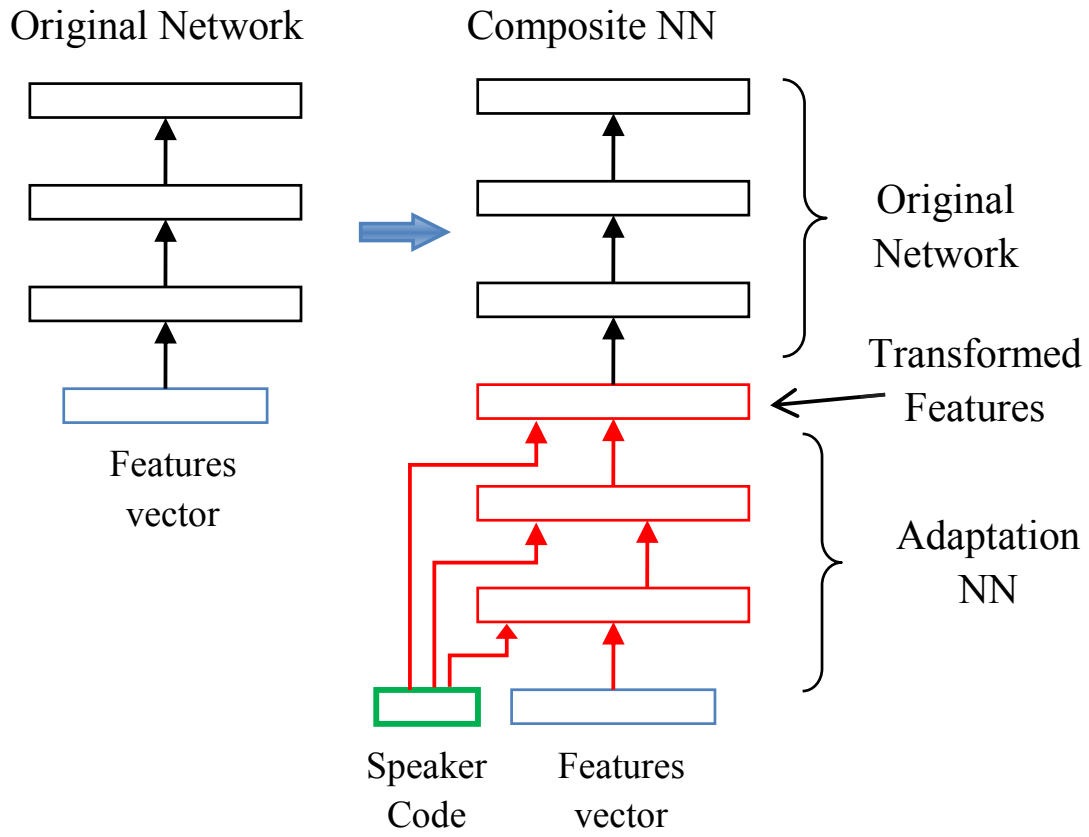

~6000 CD phone outputs
~2000 hidden units
3-8 hidden layers
~2000 hidden units
Transformed inputs
Linear input network
9x39 MFCC inputs

# LIN


~6000 CD phone outputs
~2000 hidden units
3-8 hidden layers
Fixed
~2000 hidden units
Transformed inputs
Linear input network
Adapted
9x39 MFCC inputs

# Speaker adaptation in hybrid HMM/NN systems: Speaker codes

- Basic idea: Learn a short speaker code vector for each talker



# Speaker adaptation in hybrid HMM/NN systems: i-vectors

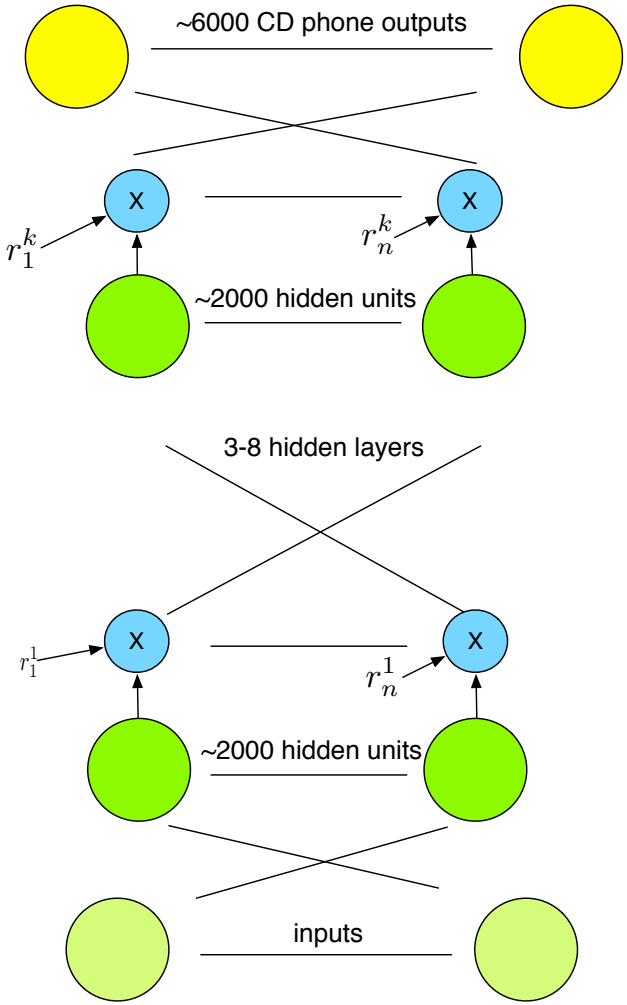
- Basic idea: Use i-vectors (speaker identity vectors) as speaker code
- i-vectors are fixed-dimensional representations $\lambda_s$, representing speaker $s$
  - a GMM is trained on all the data
  - i-vector is used (along with a matrix $M$ sometimes called the "total variability matrix") to model the difference between the means trained on all data $\mu_0$ and the speaker specific mean $\mu_s$

$$\mu_s = \mu_0 + M\lambda_s$$

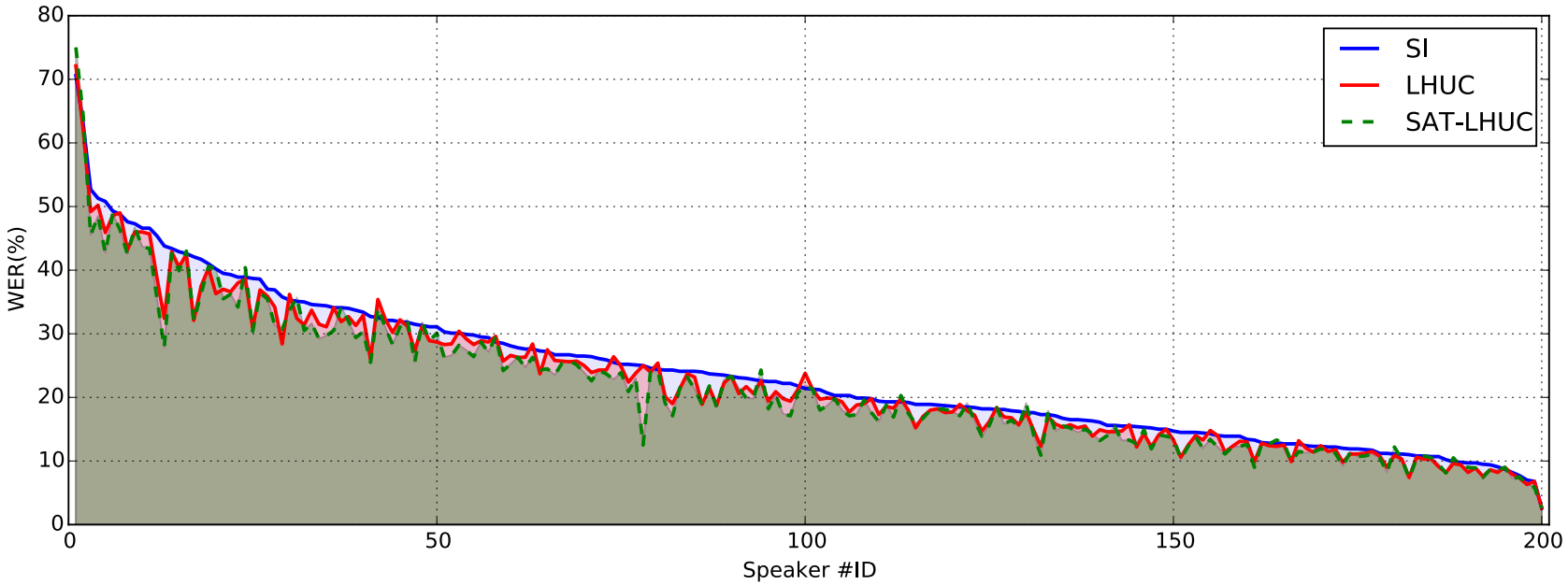
  - use an EM style algiorithm in which $\lambda_s$ and $M$ are alternatively estimated
  - this is a factor analysis approach which was developed for speaker identification
- i-vectors typically 40-100 dimensions

# Speaker adaptation in hybrid HMM/NN systems: LHUC – Learning Hidden Unit Contributions

- Basic idea: Add a learnable speaker dependent amplitude to each hidden unit
- Speaker independent: amplitudes set to 1
- Speaker dependent: learn amplitudes from data, per speaker

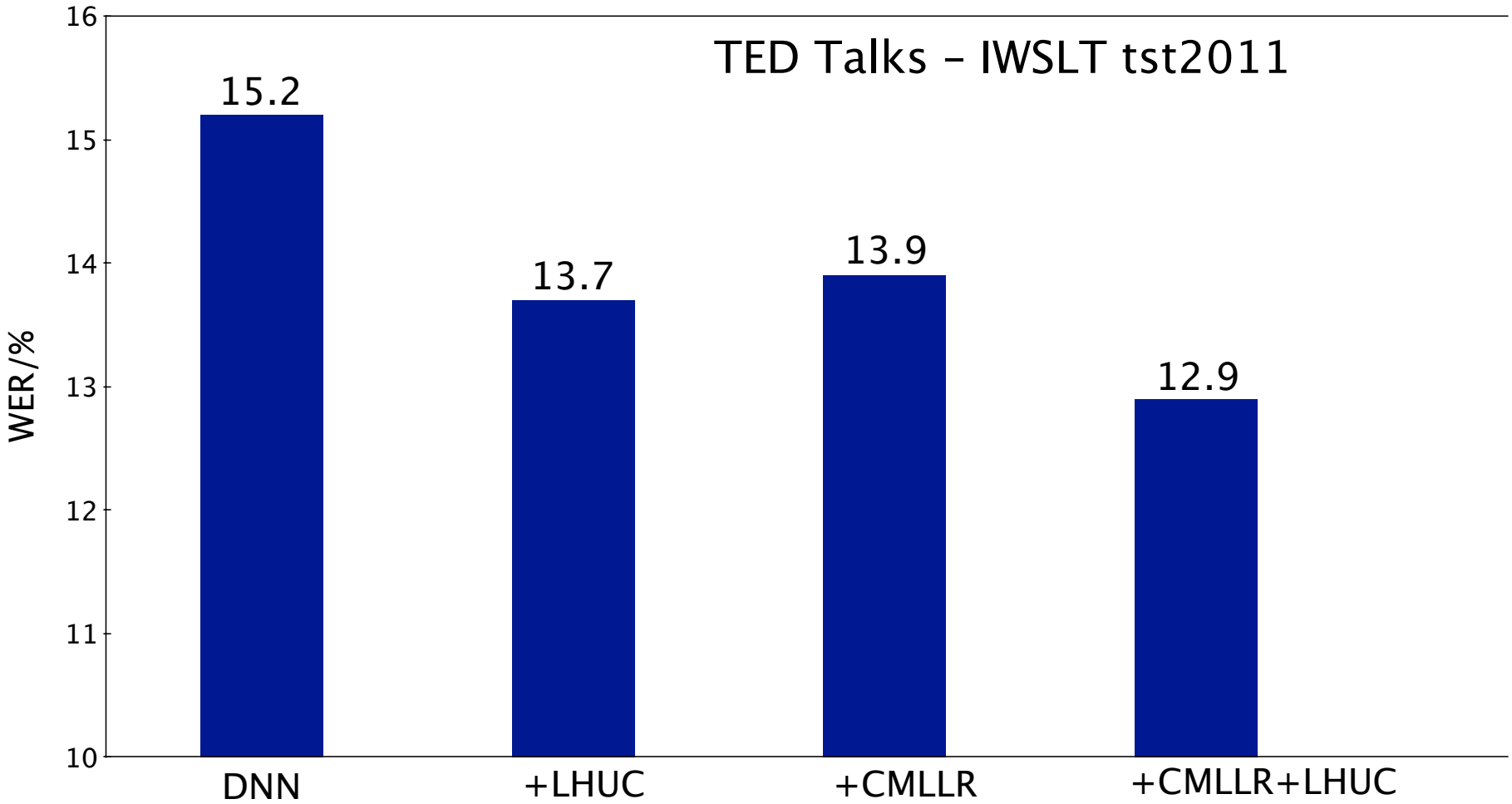


# LHUC adaptation by speaker




Results on speakers across AMI, TED, Switchboard corpora

# Speaker adaptation in hybrid HMM/NN systems: Experimental Results on TED



# Summary

## Speaker Adaptation

- One of the most intensive areas of speech recognition research since the early 1990s
- HMM/GMM
  - Substantial progress, resulting in significant, additive, consistent reductions in word error rate
  - Close mathematical links between different approaches
  - Linear transforms at the heart of many approaches
  - MLLR family is very effective
- HMM/NN
  - Open research topic
  - GMM-based feature space transforms (cMLLR) and LHUC are effective (and complementary)

# Reading


- Gales and Young (2007), "The Application of Hidden Markov Models in Speech Recognition", *Foundations and Trends in Signal Processing*, **1** (3), 195–304: section 5. http://mi.eng.cam.ac.uk/~sjy/papers/gayo07.pdf
- Woodland (2001), "Speaker adaptation for continuous density HMMs: A review", ISCA ITRW on Adaptation Methods for Speech Recognition. http://www.isca-speech.org/archive_open/archive_papers/adaptation/adap_011.pdf
- Abdel-Hamid and Jiang (2013), "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code", ICASSP-2013. https://dx.doi.org/10.1109/ICASSP.2013.6639211
- Saon et al (2013), "Speaker adaptation of neural network acoustic models using i-vectors", ASRU-2013. https://doi.org/10.1109/ASRU.2013.6707705
- Swietojanski and Renals (2014), "Learning Hidden Unit Contributions for Unsupervised Acoustic Model Adaptation", IEEE Trans. ASLP, 24:1450–1463. http://www.research.ed.ac.uk/portal/files/25128271/lhuc_final_taslp16.pdf