Speaker Adaptation

Peter Bell

Automatic Speech Recognition – ASR Lecture 13
4 March 2024

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 much lower word error rates 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 trained 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 trained 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
  - **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
  - i-vectors/speaker codes
  - Cluster-adaptive training
  - Eigenvoices

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
  - **i-vectors/speaker codes**
  - Cluster-adaptive training
  - Eigenvoices

Model-based adaptation: MAP training of GMMs

- Basic idea MAP training balances the parameters estimated on the SI data with estimates from the new data
- Consider the mean of the mth Gaussian in the jth state, $\boldsymbol{\mu}_{mj}$
  - ML estimate 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 of GMMs

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

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

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

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

    - τ controls balances the SI estimate and the adaptation data (typically $0 \leq \tau \leq 20$)
    - x_n is the adaptation vector at time n
    - $\gamma(n)$ the probability of this Gaussian at this time

Model-based adaptation: MAP training of GMMs

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

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

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

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

    - τ controls balances the SI estimate and the adaptation data (typically $0 \leq \tau \leq 20$)
    - 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, MAP estimate converges to ML estimate

The MLLR family (adapting GMMs)

- Basic idea Rather than directly adapting the model parameters, learn a transform to apply to the Gaussian means and covariances
- Problem: 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 (or all) of 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

$$\hat{\boldsymbol{\mu}} = \mathrm{A}\boldsymbol{\mu} + \mathrm{b}$$

  If the observation vectors are d-dimension, then $\boldsymbol{A}$ is a $d \times d$ matrix and b is d-dimension vector

- If we define $\mathrm{W} = [\mathrm{bA}]$ and $\boldsymbol{\eta} = [1\boldsymbol{\mu}^T]^T$, then we can write:

$$\hat{\boldsymbol{\mu}} = \mathrm{W}\boldsymbol{\eta}$$

- In MLLR, W is estimated so as to maximize the likelihood of the adaptation data
- A single transform 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 setting it to optimize the log likelihood
- Mean adaptation: Log likelihood

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

  where r ranges over the components belonging to the regression class and K_r is a constant not dependent on W
- Differentiating L and setting to 0 results in an equation for 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 can result 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

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

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

$$L = \mathcal{N}(\mathrm{A}x_n + \mathrm{b}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \log(|\mathrm{A}|) \qquad \mathrm{A}' = \mathrm{A}^{-1}; \ \mathrm{b}' = \mathrm{Ab}$$

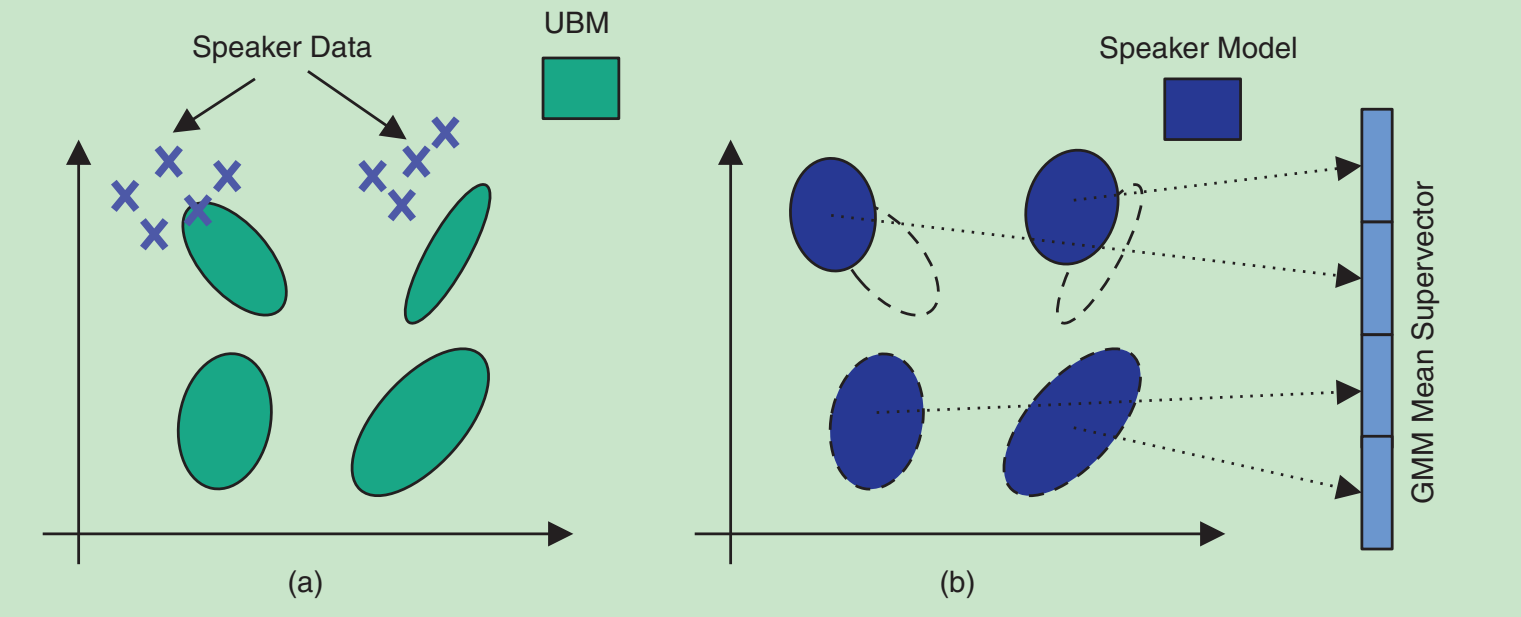
Equivalent to applying the linear transform to the data!
Also called *fMLLR (feature space MLLR)*

- Similar improvement in accuracy to standard MLLR
- Can be used as model-based feature normalisation in which the features can then be used in any system

Speaker-adaptive training (SAT)

- Basic idea Rather than SI seed (canonical) models, construct models designed for adaptation – adapt the base models to the training speakers while training
  - 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

GMM UBM system



Source: Hansen and Hasan, 2015

i-Vectors

- Represent a speaker using the GMM (mean) parameters – concatenate the target speaker mean parameters to form a **GMM supervector** $\boldsymbol{m}_s$. Typical dimension of a UBM GMM is 2048, so with 39-dimension parameters, this can be a very high dimension vector ($\sim 80,000$ components)
- Represent the supervector for an utterance $\boldsymbol{X}_u$ as the combination of the UBM supervector and the utterance **i-vector** (Dehak et al, 2011):

$$\boldsymbol{m}_u = \boldsymbol{m}_0 + \boldsymbol{T}\boldsymbol{w}_u$$

  - $\boldsymbol{m}_u$ and $\boldsymbol{m}_0$ are D-dimension supervectors for the utterance u and the UBM
  - $\boldsymbol{w}_u$ is the **i-vector** ("identity vector") – a reduced dimension (d) representation for utterance u ($d \sim 400$)
  - $\boldsymbol{T}$ is a $D \times d$ matrix (sometimes called the "total variability matrix") which projects the supervector down to the i-vector representation
  - Estimate $\boldsymbol{T}$ for the development corpus using an EM algorithm

Adapting hybrid HMM/NN systems

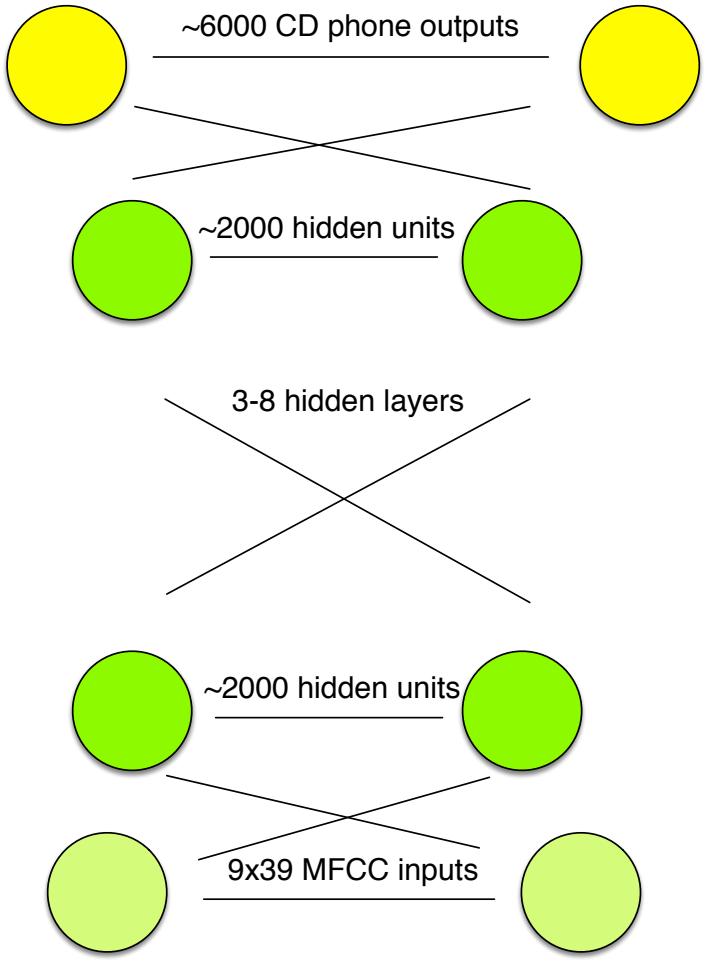
Speaker adaptation in hybrid HMM/NN systems: CMLLR feature transformation

- 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 transform)
- Limited to a single transform (regression class)
- Does not require any modification to the parameters of the DNN itself

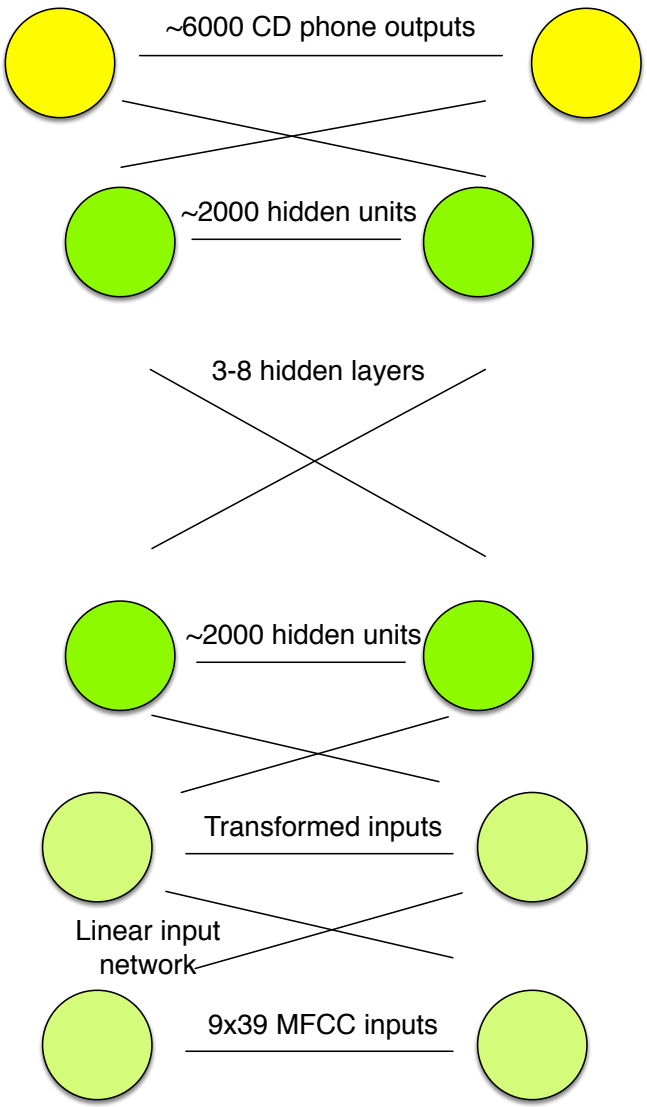
Speaker adaptation in hybrid HMM/NN systems: LIN – Linear Input Network

- 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

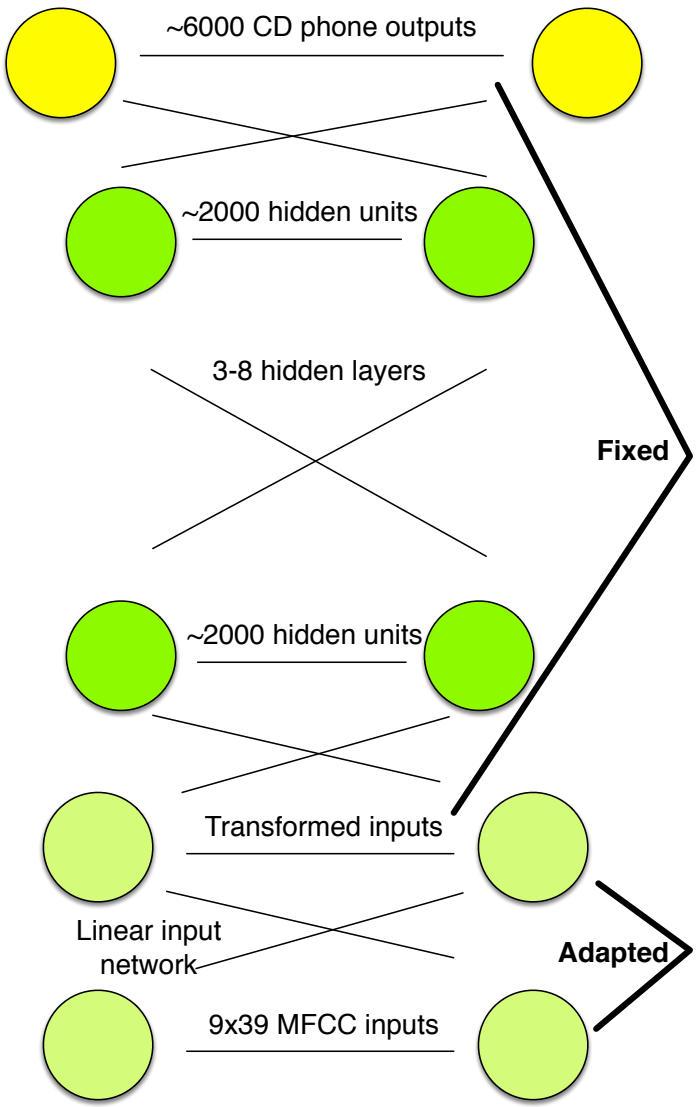
LIN


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

LIN

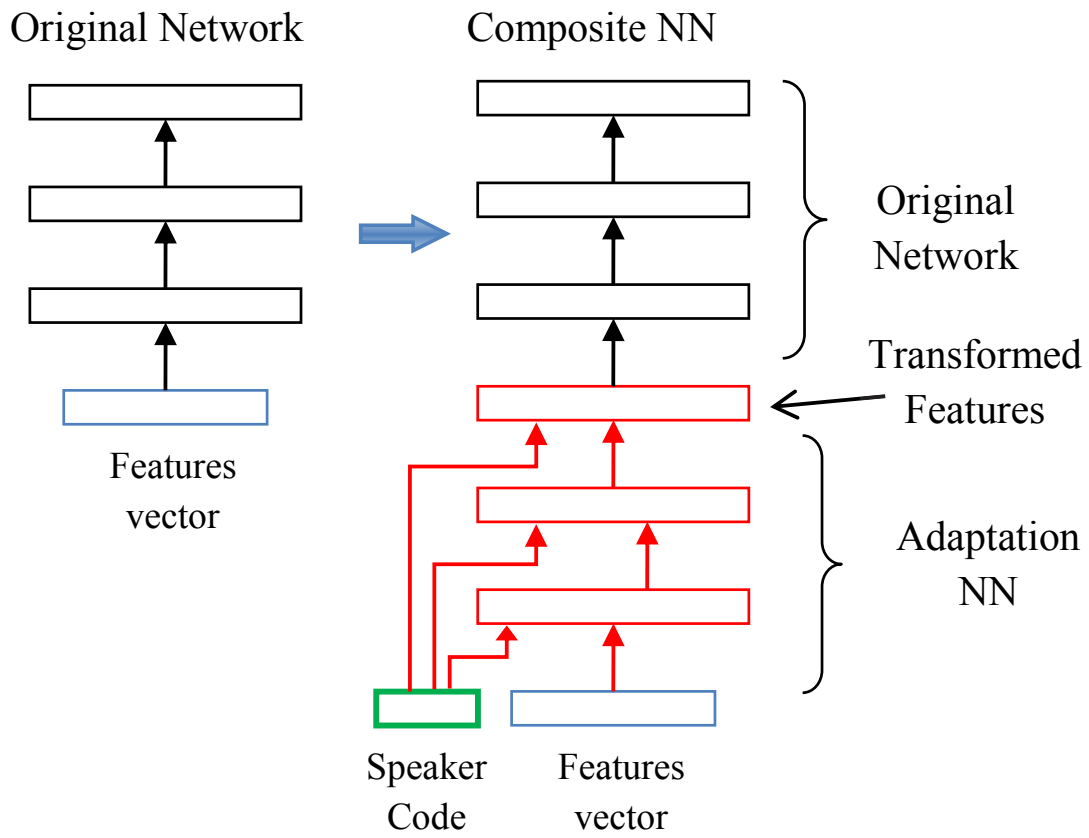

~6000 CD phone outputs
~2000 hidden units
3-8 hidden layers
~2000 hidden units
Transformed inputs
Linear input network
9x39 MFCC inputs

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

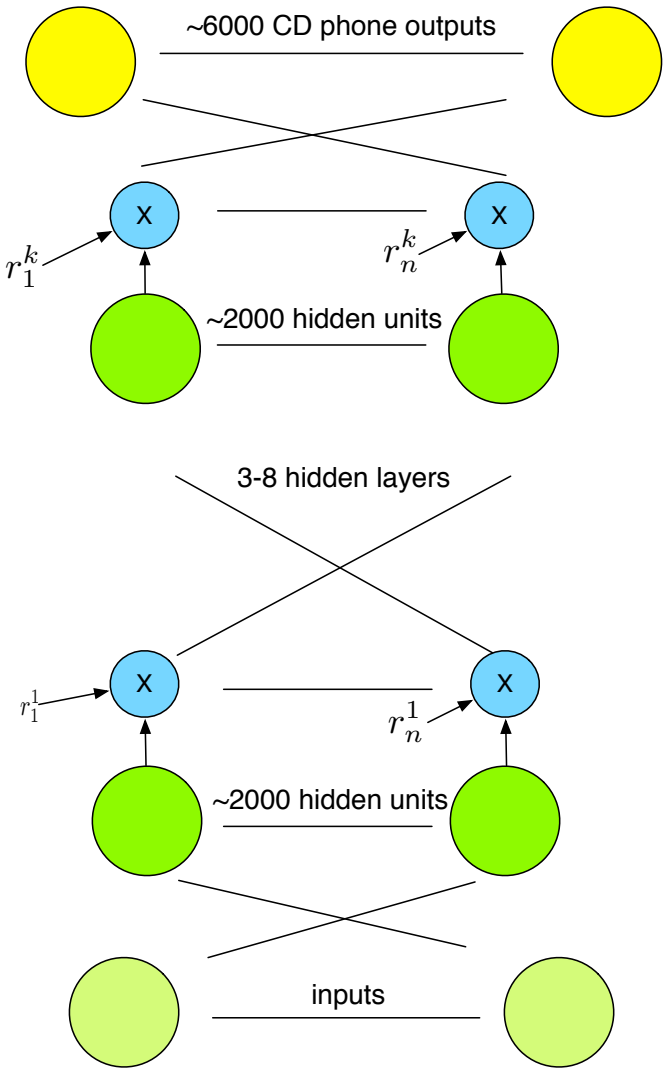


Speaker adaptation in hybrid HMM/NN systems: i-vectors

- Basic idea: Use i-vectors (speaker identity vectors) as speaker code
- Allows model to be applied to unseen speakers without further training
- i-vectors are λ_s are extracted for each training speaker
  - extracting multiple i-vectors per speaker adds robustness
- i-vectors are extracted using all available data for each test speaker
- highly effective for **dynamic** speaker adaptation

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



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

Adaptation	WER/%
None	15.7
CMLLR	15.0
+LHUC	14.4
+i-vector	14.8
+LHUC	14.2

Samarakoon and Sim (2016), "On combining i-vectors and discriminative adaptation methods for unsupervised speaker normalization in DNN acoustic models", ICASSP.
https://ieeexplore.ieee.org/abstract/document/7472684

Many other methods...

Other methods include:

- KL-divergence regularisation
- Structured transforms
- Multi-task learning
- Data augmentation
- Adversarial training

See Bell et al (2021), *Adaptation Algorithms for Neural Network-Based Speech Recognition: An Overview*

Summary

Speaker Adaptation

- An intensive area 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), i-vectors, 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
- Bell et al (2021), "Adaptation Algorithms for Neural Network-Based Speech Recognition: An Overview" , IEEE Open Journal of Signal Processing, 2, 33-36.
  https://ieeexplore.ieee.org/document/9296327/