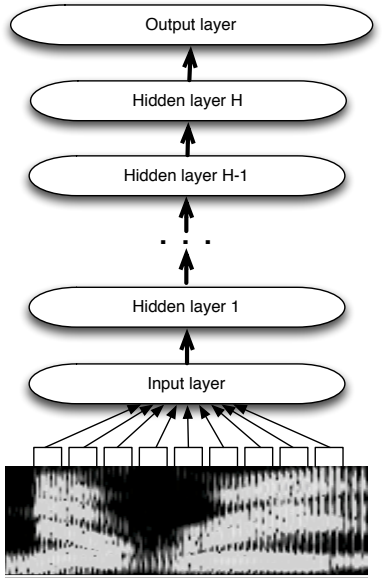
(Deep) Neural Networks

Steve Renals

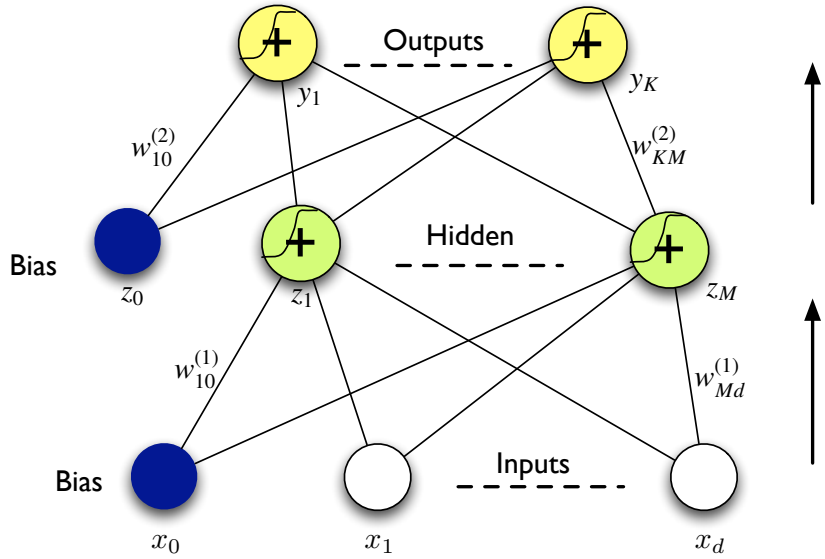
Automatic Speech Recognition— ASR Lectures 14&15
11, 18 March 2013

Neural network acoustic models



- Input layer takes several consecutive frames of acoustic features
- Output layer corresponds to classes (e.g. phones, HMM states)
- Multiple non-linear hidden layers between input and output
- Neural networks also called multi-layer perceptrons

Layered neural networks: structure (1)



Neural network with one hidden layer

Layered neural networks: structure (2)

- d input units, M hidden units, K output units
- Hidden layer: each of M units takes a linear combination of the inputs x_i:

$$b_j = \sum_{i=0}^{d} w^{(1)}_{ji} x_i$$

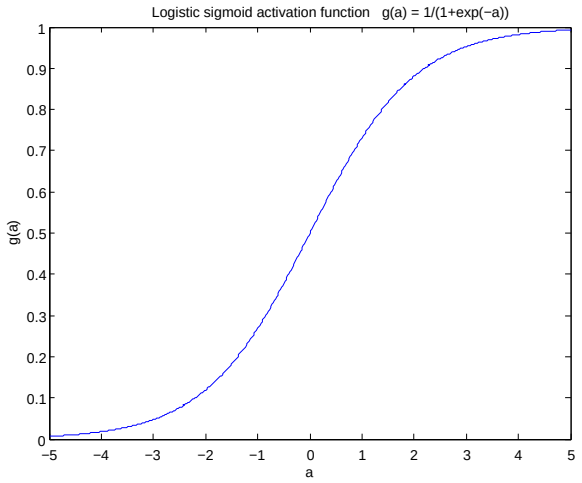
  - b_j: activations
  - $w^{(1)}_{ji}$: first layer of weights
- Activations transformed by a nonlinear activation function h (e.g. a sigmoid):

$$z_j = h(b_j) = \frac{1}{1 + \exp(-b_j)}$$

  - z_j: hidden unit outputs

Logisitic sigmoid activation function

$$g(a) = \frac{1}{(1 + \exp(-a))}$$



Layered neural networks: structure (3)

- Outputs of the hidden units are linearly combined to give activations of the output units:

$$a_k = \sum_{j=0}^{M} w_{kj}^{(2)} z_j$$

- The output units are transformed using the softmax activation function:

$$y_k = g(a_k) = \frac{\exp a_k}{\sum_{\ell=1}^{K} \exp(a_\ell)}$$

- If output unit k corresponds to class C_k, then interpret outputs of trained network as posterior probability estimates

$$P(C_k \mid \mathbf{x}) = y_k$$

Layered neural networks: training

- Weights w_{ji} are the trainable parameters
- Train the weights by adjusting them to minimise a cost function which measures the error the network outputs y_k (per frame) compared with the *target* output t_k
- Cross-entropy between actual and target outputs

$$E = -\sum_{k=1}^{K} t_k \log y_k$$

- Optimise the cost function using *gradient descent* (back-propagation of error — backprop)

Gradient descent

- Gradient descent can be used whenever it is possible to compute the derivatives of the error function E with respect to the parameters to be optimized $\mathbf{W}$
- Basic idea: adjust the weights to move downhill in *weight space*
- Weight space: space defined by all the trainable parameters (weights)
- Operation of gradient descent:
  1. Start with a guess for the weight matrix $\mathbf{W}$ (small random numbers)
  2. Update the weights by adjusting the weight matrix in the direction of $-\nabla_{\mathbf{W}} E$.
  3. Recompute the error, and iterate
- The update for weight w_{ki} at iteration $\tau + 1$ is:

$$w_{ki}^{\tau+1} = w_{ki}^{\tau} - \eta \frac{\partial E}{\partial w_{ki}}$$

The parameter η is the *learning rate*

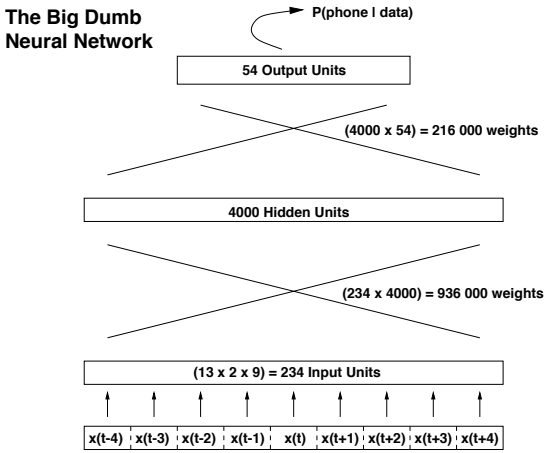
HMM/NN hybrid systems

- Basic idea: in an HMM, replace the GMMs used to estimate output pdfs with the outputs of neural networks
- Transform NN posterior probability estimates to *scaled likelihoods* by dividing by the relative frequencies in the training data of each class

$$P(\mathbf{x}_t \mid C_k) \propto \frac{P(C_k \mid \mathbf{x}_t)}{P_{\text{train}}(C_k)} = \frac{y_k}{P_{\text{train}}(C_k)}$$

- NN outputs correspond to phone classes or HMM states

Monophone HMM/NN hybrid system (1990s) (1)



- Similar performance to context-dependent HMM/GMM systems on WSJ
- More errors on more complex tasks (broadcast news, conversational telephone speech)

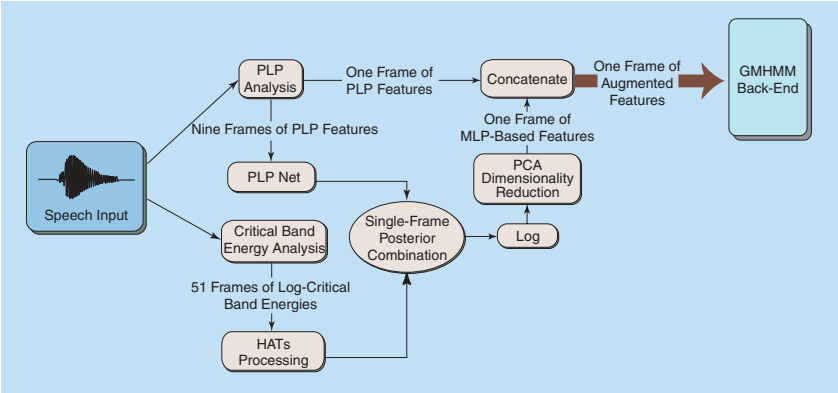
HMM/NN vs HMM/GMM

- Advantages of NN:
  - Incorporate multiple frames of data at input
  - More flexible than GMMs (i.e. not made of (nearly) local components) — GMMs inefficient for non-linear class boundaries
- Disadvantages of NN:
  - Context-independent (monophone) models
  - Weak speaker adaptation algorithms
  - Computationally expensive - more difficult to parallelise than GMM systems
- Reading: Morgan and Bourlard (1995)

Tandem features (posteriorgrams)

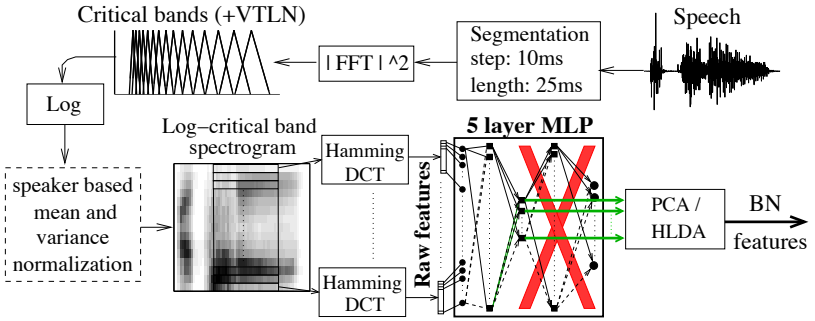
- Use NN probability estimates as an additional input feature stream in an HMM/GMM system —- (*Tandem* features (i.e. NN + acoustics), posteriorgrams)
- Advantages of tandem features
  - can be estimated using a large amount of temporal context (eg up to ± 25 frames)
  - encode phone discrimination information
  - only weakly correlated with PLP or MFCC features
- Tandem features: reduce dimensionality of NN outputs using PCA, then concatenate with acoustic features (e.g. MFCCs)

Tandem features



Morgan et al (2005)

Bottleneck features

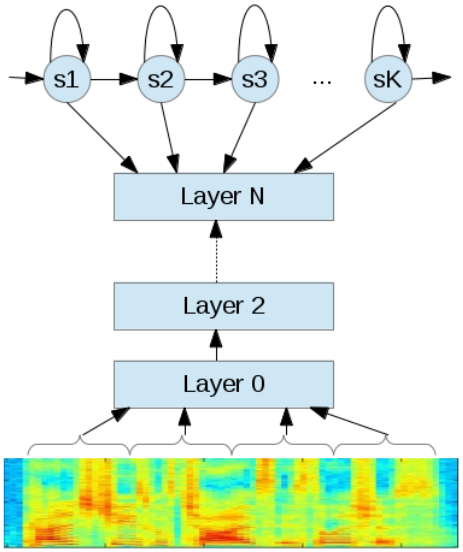


- Grezl and Fousek (2008)
- Use a "bottleneck" hidden layer to provide features for a HMM/GMM system

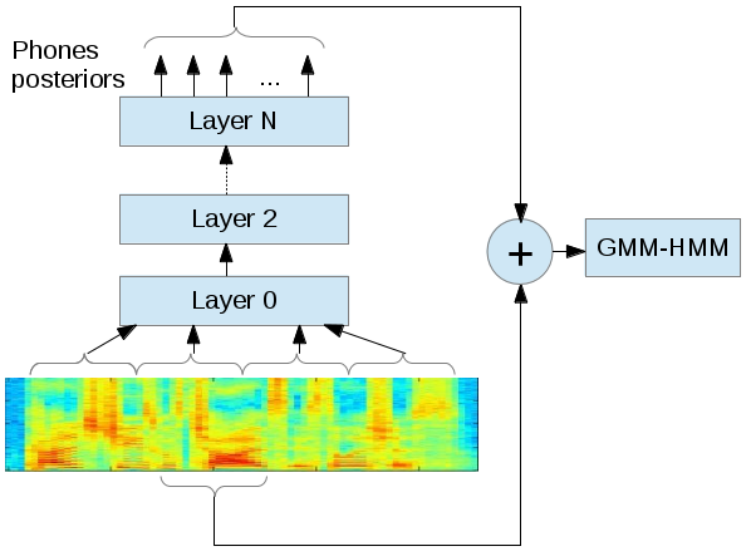
Interim summary

- Hybrid HMM/NN system can be powerful acoustic models — but unadapted monophone NN-based system have worse accuracies than state-of-the-art GMM systems on complex tasks
- Using NNs to provide tandem features (posteriorgrams, bottleneck features) for GMMs can significantly reduce word error rates (10-15%)

Deep neural networks (DNNs) — Hybrid system



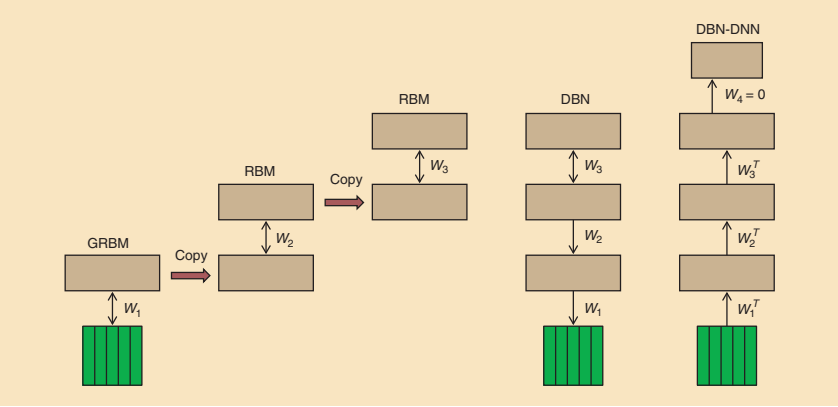
Deep neural networks (DNNs) — Tandem system



DNNs — what's new?

- Training multi-hidden layers directly with gradient descent is difficult — sensitive to initialisation, gradients can be very small after propagating back through several layers. *Unsupervised pretraining*(see Hinton et al 2012)
  - Train a stacked restricted Boltzmann machine generative model (unsupervised), then finetune with backprop
  - Contrastive divergence training
- Many hidden layers
  - GPUs provide the computational power
- Wide output layer (context dependent phone classes)
  - GPUs provide the computational power

Unsupervised pretraining



Hinton et al (2013)

Example: hybrid HMM/DNN phone recognition (TIMIT)

- Train a 'baseline' three state monophone HMM/GMM system (61 phones, 3 state HMMs) and Viterbi align to provide DNN training targets (time state alignment)
- The HMM/DNN system uses the same set of states as the HMM/GMM system — DNN has 183 (61*3) outputs
- Hidden layers — many experiments, exact sizes not highly critical
  - 3–8 hidden layers
  - 1024–3072 units per hidden layer
- Multiple hidden layers always work better than one hidden layer
- Pretraining always results in lower error rates
- Best systems have lower phone error rate than best HMM/GMM systems (using state-of-the-art techniques such as discriminative training, speaker adaptive training)

Example: hybrid HMM/DNN large vocabulary conversational speech recognition (Switchboard)

- Recognition of American English conversational telephone speech (Switchboard)
- Baseline context-dependent HMM/GMM system
  - 9,304 tied states
  - Discriminatively trained (BMMI — similar to MPE)
  - 39-dimension PLP (+ derivatives) features
  - Trained on 309 hours of speech
- Hybrid HMM/DNN system
  - Context-dependent — 9304 output units obtained from Viterbi alignment of HMM/GMM system
  - 7 hidden layers, 2048 units per layer
- DNN-based system results in significant word error rate reduction compared with GMM-based system
- Can also use DNNs in tandem configuration (next lecture)

Speaker adaptation in hybrid systems: LIN

- 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

Speaker adaptation in hybrid systems: CMLLR

- 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)

Reading

- N Morgan and H Bourlard (May 1995). "Continuous speech recognition: An introduction to the hybrid HMM/connectionist approach", *IEEE Signal Processing Magazine*, **12**(3), 24–42.
- N Morgan et al (Sep 2005). "Pushing the envelope — aside", *IEEE Signal Processing Magazine*, **22**(5), 81–88.
- F Grezl and P Fousek (2008). "Optimizing bottleneck features for LVCSR", Proc ICASSP–2008.
- G Hinton et al (Nov 2012). "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups", *IEEE Signal Processing Magazine*, **29**(6), 82–97.