Deep Learning for Automatic Speech Recognition

CSCI/LING-5832: Natural Language Processing

Ryan Hartsfield Garrett Lewellen

April 23, 2015

Outline

Speech Recognition

Main goal of any automatic speech recognition system is to take some acoustic signal as input and convert it into a corresponding lexical representation.

Tasks range from very easy to extremely difficult: Number recognition, phone recognition, large vocabulary speech recognition, human-computer, human-human

Many variable factors to account for: environmental noise, speaker variation, vocabulary size, degree of fluency, etc.

The ASR Problems

There are **four** substantial problems that must be solved in any statistical ASR system:

1. Acoustic Processing
2. Acoustic Modeling
3. Language Modeling
4. The Search Problem

Acoustic Processing

Represent acoustic data in a way that has few model parameters, but maintains necessary linguistic information.

Main form of feature representation are the mel-frequency cepstral coefficients (MFCCs) extracted from a waveform.

- Convert the spectrum of a window into its cepstrum
- Effectively separates the source from filter
- Cepstral coefficients are uncorrelated
- Use the first 12 (represent the filter), their deltas and delta-deltas for 39 features per frame

$$c[n] = \sum_{n=0}^{N-1} \log(|\sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}|) e^{j\frac{2\pi}{N}kn}$$

Acoustic Modeling

Given the observation feature vector, estimate an HMM state posterior probability:

$$\mathbb{P}(x|w)$$

Where x is some acoustic signal as input and w is a word, phone, sub-phone, diphone, or triphone.

We will explore three ways of estimating this probability:

- Gaussian Mixture Models (GMMs)
- Convolutional Neural Networks (CNNs)
- Deep Neural Networks (DNNs)

Language Modeling and Search

Job of the language model is to estimate the prior probability:
$\mathbb{P}(w)$

This is usually done with an N-gram model like we've seen throughout the semester.

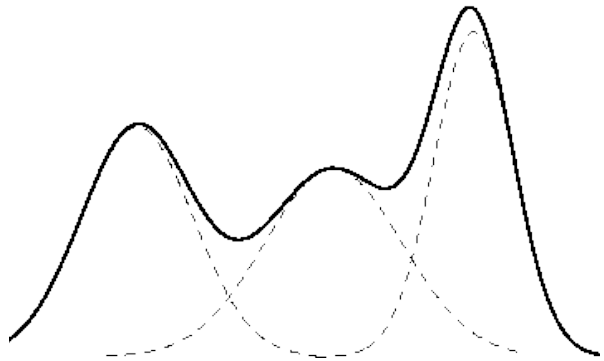
We normally use GMM-HMM or ANN-HMM hybrids for acoustic/language models. Given these models, we need to find the best sequence of words for the acoustic input.

Search is normally performed using Viterbi, A* decoding, or N-best.

Outline

Gaussian Mixture Models

TODO: clean up



Before the rise of Deep Learning approaches, GMMs were ubiquitous for representing the acoustic model.

Based on assumption that a mixture of Gaussians can be used to represent the spectral shape.

GMM Models

Use a weighted mixture of multivariate Gaussians as a model to assign a likelihood to an acoustic observation.

Below is the output likelihood function $b_j(o_t)$:

$$b_j(o_t) = \sum_{m=1}^{M} c_{jm} \frac{1}{\sqrt{2\pi|\sigma_{jm}|}} \exp[(x - \mu_{jm})^T \sigma_{jm}^{-1}(o_t - \mu_{jm})]$$

A GMM model can be trained using the Baum-Welch algorithm to iteratively recompute the mean, mixture weight, and covariance.

The Rise of Deep Learning

Recently, Artificial Neural Networks (ANNs) have experienced great results in a variety of tasks, and ANN-HMMs have begun to replace GMM-HMMs.

- Increasing processing power and more efficient algorithms for training.
- GMMs have much fewer model parameters, and are much easier to train.
- ANN-HMM hybrids are much larger, and can take in multiple frames (9-15)
- GMM-HMMs typically only look at a single frame.

This increase in size allows hybrid ANN-HMMs to model highly correlated features across wide temporal contexts.

Outline

Convolutional Neural Networks for Speech Recognition

Ossama Abdel-Hamid
Li Deng
Abdel-rahman Mohamed
Gerald Penn
Hui Jiang
Dong Yu

IEEE/ACM Transactions on Audio, Speech, and Language processing, Vol. 22, No. 10, October 2014

CNN Background

CNNs have achieved high levels of success in image recognition tasks. They have a very complex architecture that is influenced by our own visual cortex.

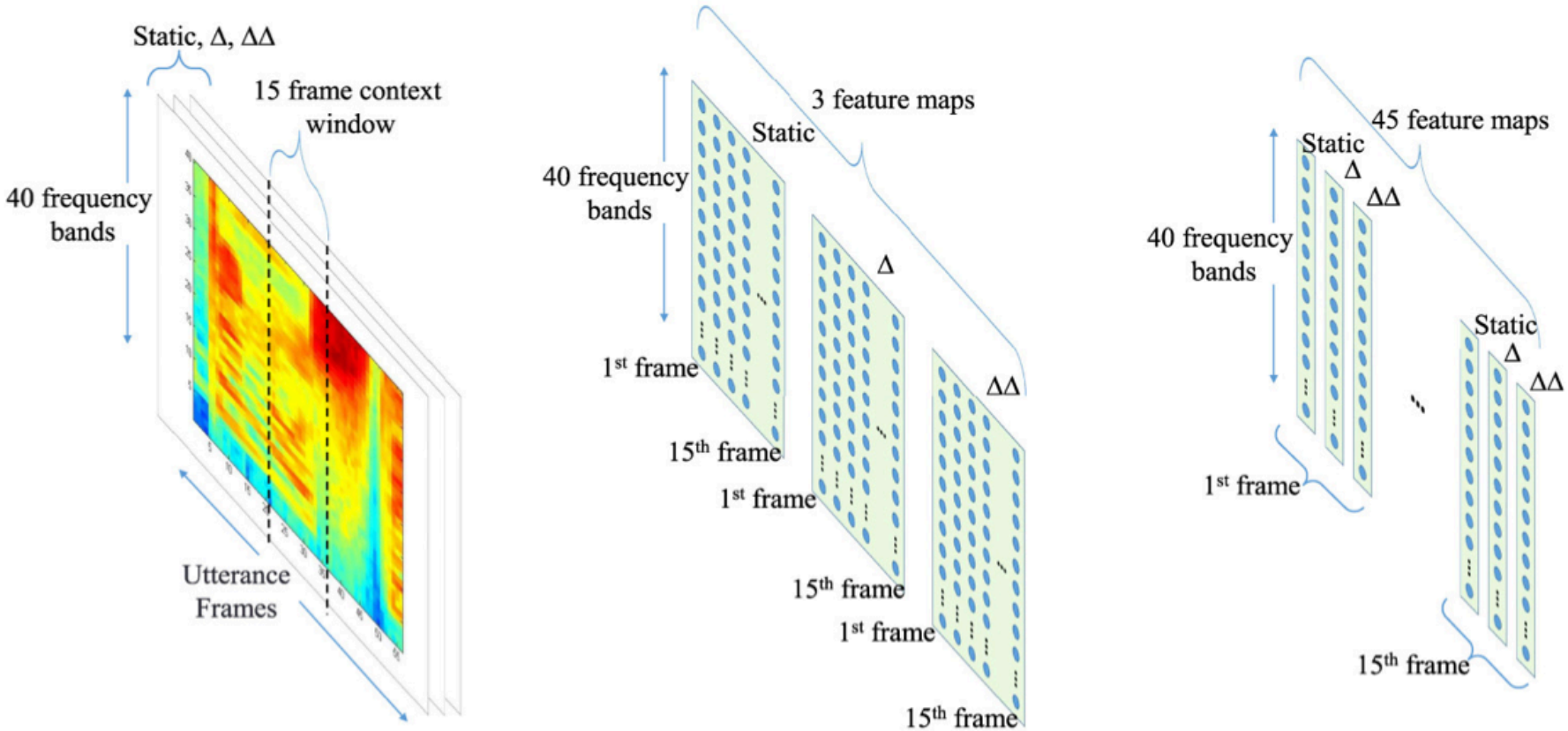
- Special layer with a convolution ply and pooling ply (simple cells and complex cells)
- Weights are shared across all neurons in the same convolution ply
- Replicates features across the image in order to recognize patterns anywhere within it

CNN-HMM for ASR

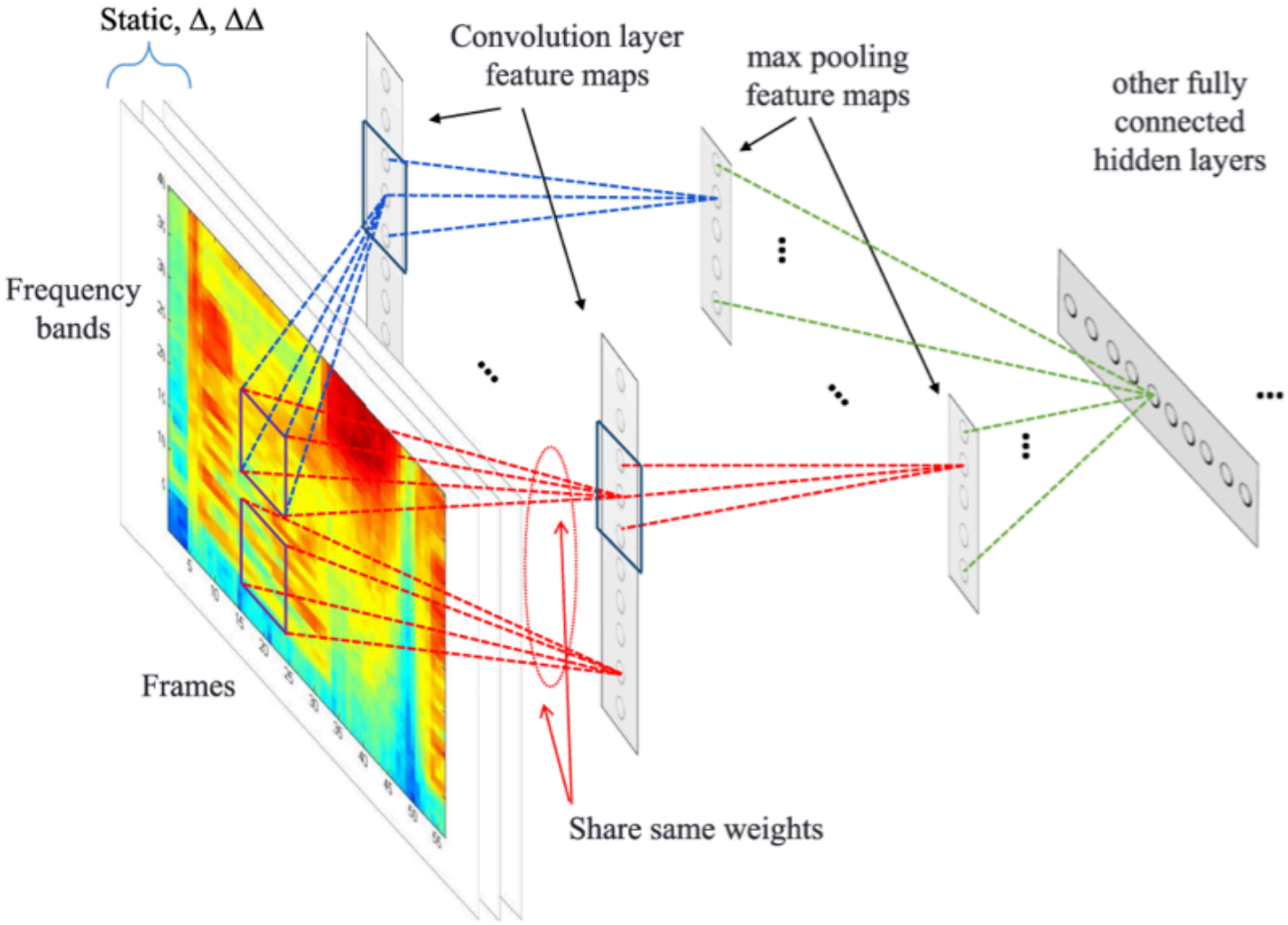
Apply a similar architecture to the task of ASR with some differences to help exhibit invariance to small perturbations in frequency caused by different speakers, emotional status, background noise, etc.:

- Local Connectivity: Mel-frequency spectral coefficients (MFSC) features
- Convolutions along frequency axis: HMMs handle time axis, allow for small frequency shifts
- Limited weight sharing: Same pool = Same weights

Input Structure of the CNN



Global Architecture



Convolution Ply

The convolution ply introduces locality into the network. Unlike in image processing, weights are only shared between neurons that go to the same pooling ply.

Limited weight sharing is used to reflect the fact that properties of the speech signal vary over different frequency bands. Each frequency band has its own shared weights.

Each unit in a single feature map can be computed as below:

$$q_{j,m} = \sigma(\sum_{i=1}^{I}\sum_{n=1}^{F} o_{i,n+m-1} w_{i,j,n} + w_{0,j}), (j = 1, ..., J)$$

Pooling Ply

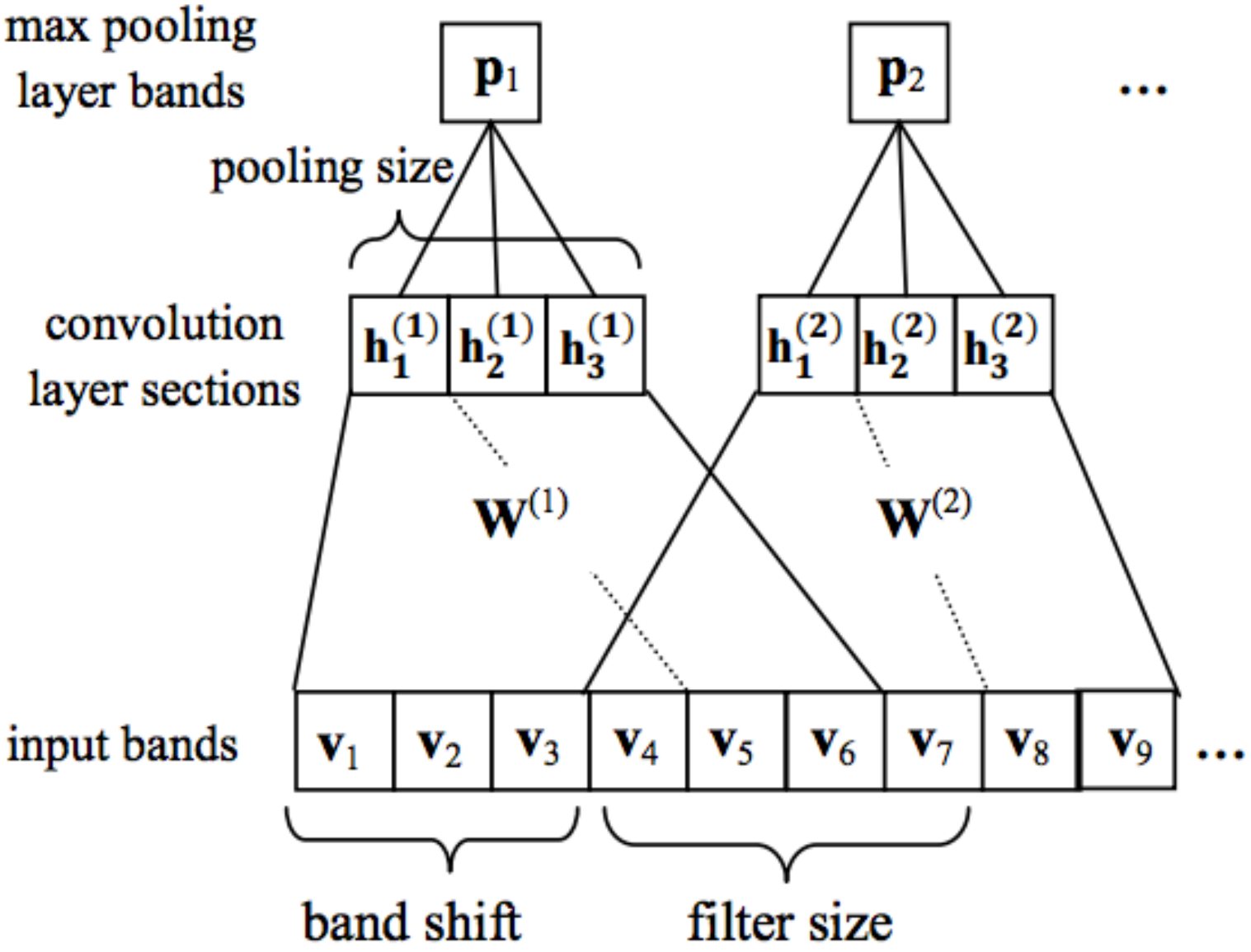
The pooling ply reduces the resolution of the feature maps, serving as a generalization over the features of the convolution ply. This helps with handling small shifts in frequency.

In this paper they use a simple max-pooling function:

$$p_{i,m} = \max_{n=1}^{G} q_{i,(m-1)xs+n}$$

where G is the pooling size and s is the shift size. If $G = s$, there is no overlap between the pooling windows.

A Simpler Example



Training with Back-Propagation

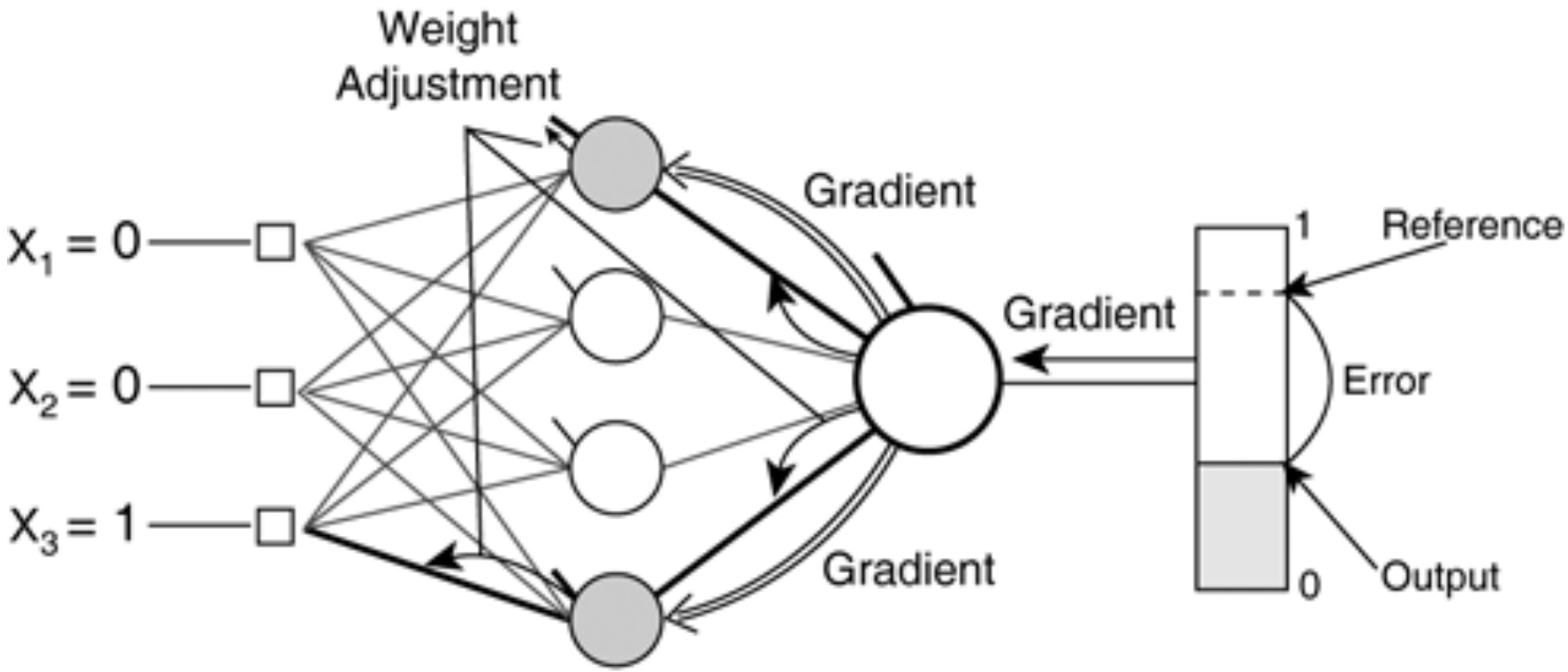
Most frequently used training algorithm for ANNs. We want to discover how the output of the ANN changes with respect to the input. Both networks use a softmax output layer to compute the posterior probability:

$$y_i = \frac{\exp(o_i^{(L)})}{\sum_j \exp(o_j^{(L)})}$$

Goal is to minimize the cross-entropy function: $Q(W^{(l)}) = -\sum_i d_i \log y_i$ where d is the target and y is the prediction.

Compute derivative of Q with respect to each weight matrix, and use stochastic gradient descent to minimize the objective function.

Back-Propagation cont.



Outline

Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition

George Dahl Dong Yi Li Deng Alex Acero

IEEE Transactions on Audio, Speech, and Language processing,
Vol. 20, No. 1, January 2012

Problem Formulation

Noisy channel model: maximize the likelihood of a decoded word sequence, $\hat{w}$, given our observed audio input, x:

$$\hat{w} = \underset{w \in \mathscr{L}}{\operatorname{argmax}} \underbrace{\mathbb{P}(x|w)}_{\text{Accoustic Model}} \underbrace{\mathbb{P}(w)}_{\text{Language Model}}$$

Use N-gram language model, CD-DNN-HMM for acoustic model

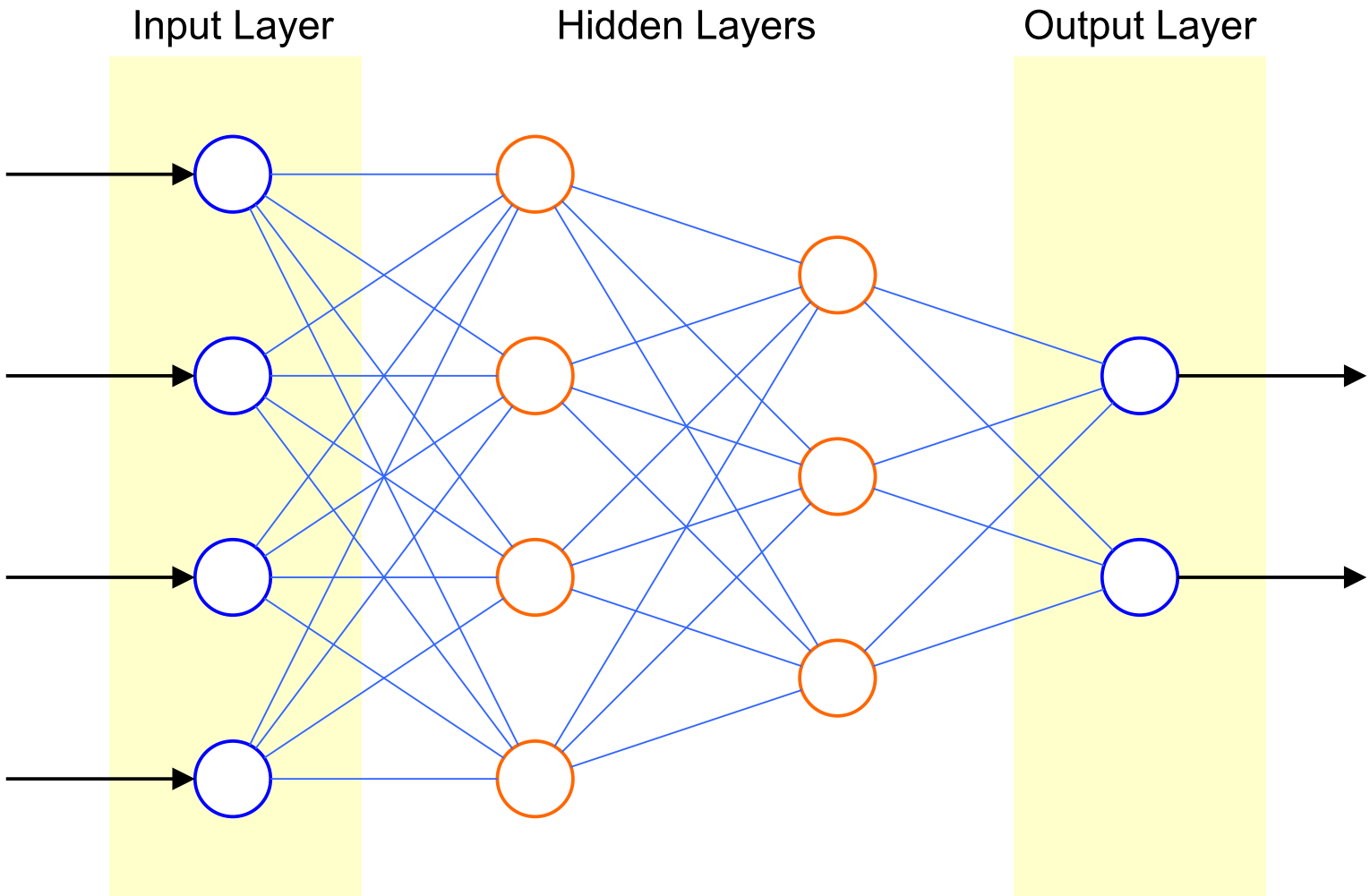
Problem Formulation (Cont)

Here the acoustic model is viewed as a sequence of transitions between states of tied-state triphones referred to as **senones**.

$$\underbrace{\mathbb{P}(x|w)}_{\text{Accoustic Model}} \approxeq \max \underbrace{\pi(q_0)}_{\text{Init State}} \prod_{t=1}^{T} \underbrace{a_{q_{t-1}q_t}}_{\text{Transistion}} \prod_{t=0}^{T} \underbrace{\mathbb{P}(x_t|q_t)}_{\text{Senone posterior}}$$

Where $\mathbb{P}(x_t|q_t)$ models the tied triphone senone posterior given mel-frequency cepstral coefficients (**MFCCs**) based on 11 sampled frames of audio.

Deep Neural Networks



Training

Training is complicated and time intensive! At a very high level two steps: initialization and DNN training.

Initialization:

- Find the best tying of triphone states
- Deal with some book keeping
- Train a CD-GMM-HMM using those states.
- Convert CD-GMM-HMM into CD-DNN-HMM keeping senone structure
- Apply pre-training algorithm on CD-DNN-HMM

Training

DNN Training:

- Generate raw alignment of states to senones
- Use alignment to refine by backpropagation
- Re-estimate prior senone probability given frames
- Refine HMM transitions probabilities
- Goto first step if no improvement against development set

Just enough time to talk about one of these steps in detail. Let's look at pre-training...

Pre-Training In-Depth

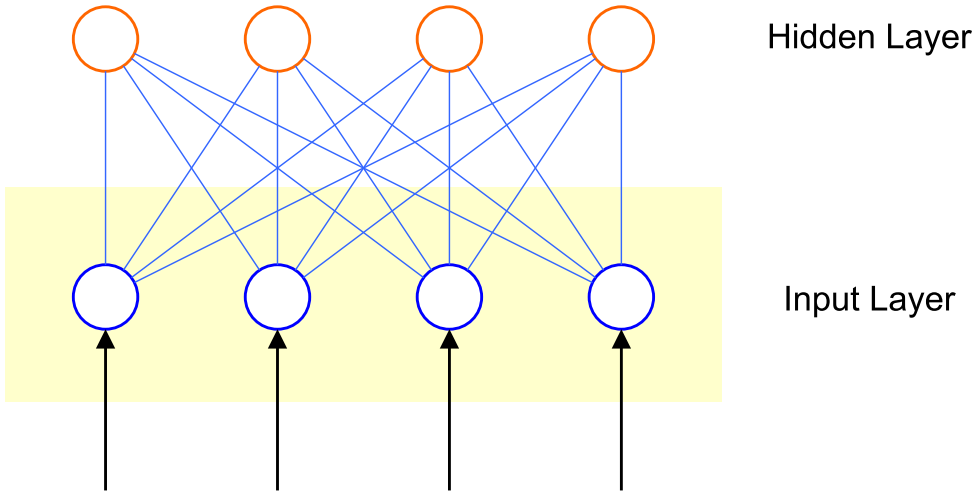
DNN training computationally intractable until Hinton et al. come to the rescue with *A Fast Learning Algorithm for Deep Belief Nets*.

Big idea: Use an approximate method, **contrastive divergence**, to get near an optimal solution, then use traditional methods, **backpropagation**, to finish the job.

Need to understand Restricted Boltzman Machines (RMBs) and Deep Belief Networks (DBNs)

Pre-Training: RBMs and DBNs

Bipartite arrangement of weights is assigned an energy:



$$E(v, h) = -b^T v - c^T h - v^T W h$$

For purpose of this talk, stack RBMs on top of one another to get a DBN

Pre-Training: Contrastive Divergence

Want to do vanilla Stochastic Gradient Descent, however, our model term takes **exponential** time to compute correctly.

$$-\frac{\partial \ell(\theta)}{\partial w_{ij}} = \left\langle \frac{\partial E}{\partial \theta} \right\rangle_{\text{data}} - \left\langle \frac{\partial E}{\partial \theta} \right\rangle_{\text{model}}$$

Instead, **approximate** it (essentially minimizing Kullback-Leibler divergence) with one step Gibbs sampling:

$$-\frac{\partial \ell(\theta)}{\partial w_{ij}} \approx \langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_1$$

Pre-Training: Bringing it all together

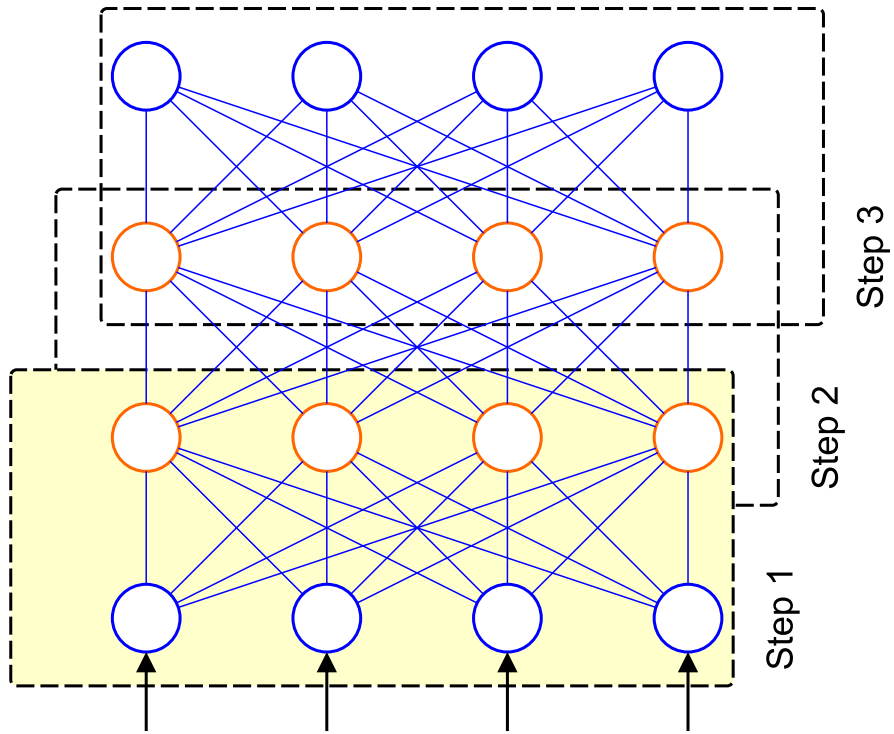
Hinton's Greedy Algorithm:

1) Train RBM consisting of first two layers

2) Move RBM frame up a layer and train

3) When out of layers, you have trained DBN

4) Refine with backpropagation



Outline

Experimental Results

Papers report different metrics (sentence accuracy, phone error rate) on different datasets (Bing, TIMIT)

	Bing Mobile		TIMIT	
Architecture	Dev.	Test	Dev.	Test
CD-GMM-HMMM	70.3%	68.4%		
CD-DNN-HMM	71.8%	69.6%		
CNN-HMM				20.07%

Take away: Both demonstrate significant improvement over GMM approach, with CNN approach giving the best performance.

Outline

Conclusions

TODO: Garrett

Conclusions (Cont)

TODO: Garrett

Questions?