

# Deep Neural Network and Its Application in Speech Recognition

---

*Dong Yu*

*Microsoft Research*

**Thanks to my collaborators:**

**Li Deng, Frank Seide, Gang Li, Mike Seltzer, Jinyu Li, Jui-Ting Huang,  
Kaisheng Yao, Jian Xue, Yan Huang, Adam Eversole, George Dahl,  
Abdel-rahman Mohamed, Xie Chen, Hang Su, Ossama Abdel-Hamid,  
Eric Wang, Andrew Maas, and many more**

# Part 3: Outline

---

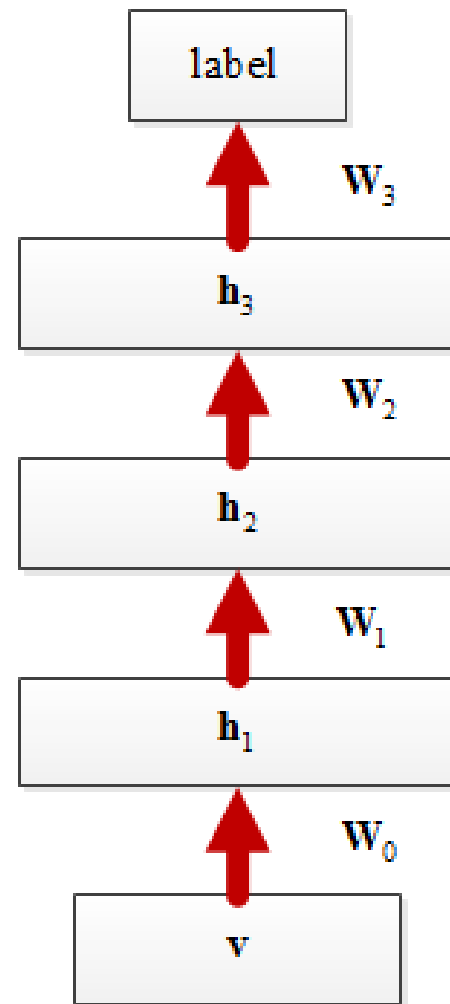
- **CD-DNN-HMM**
- Training and Decoding Speed
- Invariant Features
- Mixed-bandwidth ASR
- Multi-lingual ASR
- Sequence Discriminative Training
- Adaptation
- Summary

# Deep Neural Network

- A fancy name for multi-layer perceptron (MLP) with **many hidden layers**.
- Each sigmoidal hidden neuron follows Bernoulli distribution
- The last layer (softmax layer) follows multinomial distribution

$$p(l = k | \mathbf{h}; \theta) = \frac{\exp(\sum_{i=1}^H \lambda_{ik} h_i + a_k)}{Z(\mathbf{h})}$$

- Training can be difficult and tricky. Optimization algorithm and strategy can be important.

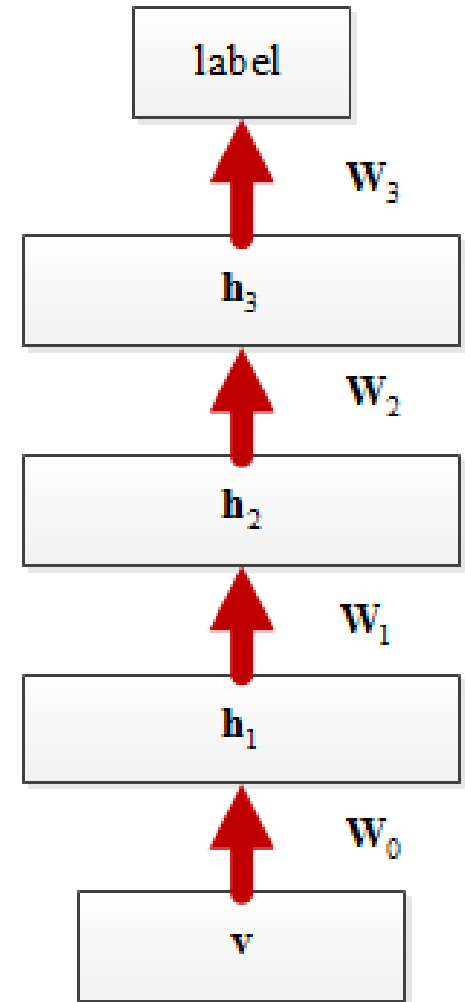


# Deep Neural Network

- A fancy name for multi-layer perceptron (MLP) with **many hidden layers**.
- Each sigmoidal hidden neuron follows Bernoulli distribution
- The last layer (softmax layer) follows multinomial distribution

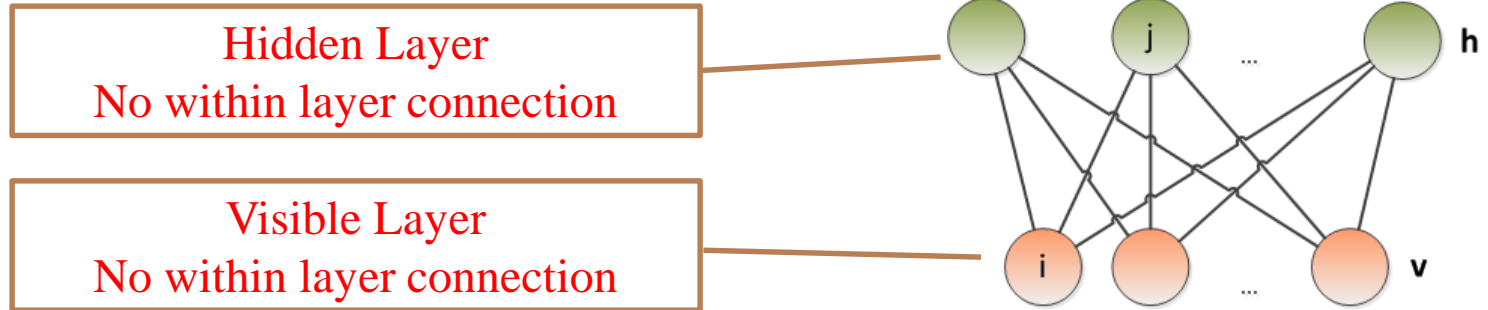
$$p(l = k | \mathbf{h}; \theta) = \frac{\exp(\sum_{i=1}^H \lambda_{ik} h_i + a_k)}{Z(\mathbf{h})}$$

- **Training can be difficult and tricky. Optimization algorithm and strategy can be important.**



# Restricted Boltzmann Machine

(Hinton, Osindero, Teh 2006)



- Joint distribution  $p(\mathbf{v}, \mathbf{h}; \theta)$  is defined in terms of an energy function  $E(\mathbf{v}, \mathbf{h}; \theta)$

$$p(\mathbf{v}, \mathbf{h}; \theta) = \frac{\exp(-E(\mathbf{v}, \mathbf{h}; \theta))}{Z}$$

$$p(\mathbf{v}; \theta) = \sum_{\mathbf{h}} \frac{\exp(-E(\mathbf{v}, \mathbf{h}; \theta))}{Z} = \frac{\exp(-F(\mathbf{v}; \theta))}{Z}$$

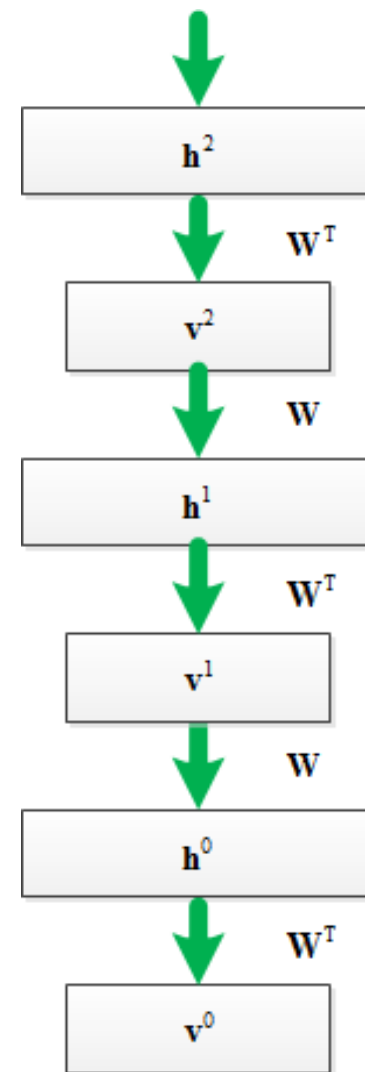
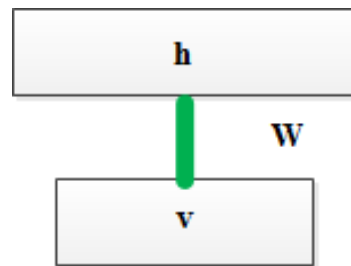
- Conditional independence

$$p(\mathbf{h}|\mathbf{v}) = \prod_{j=0}^{H-1} p(h_j|\mathbf{v})$$

$$p(\mathbf{v}|\mathbf{h}) = \prod_{i=0}^{V-1} p(v_i|\mathbf{h})$$

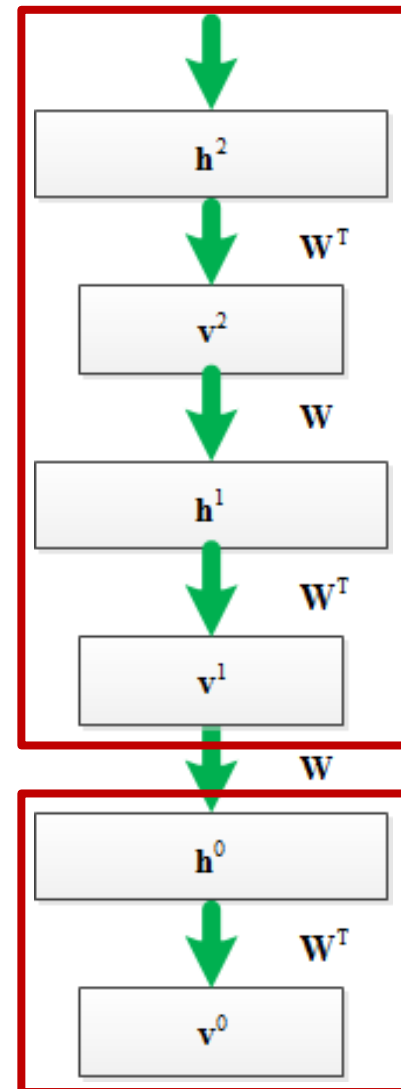
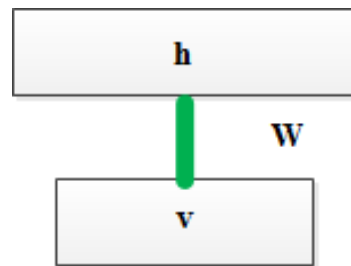
# Generative Pretraining a DNN

- First learn with all the weights tied
  - equivalent to learning an RBM
- Then freeze the first layer of weights and learn the remaining weights (still tied together).
  - equivalent to learning another RBM, using the aggregated conditional probability on  $\mathbf{h}_0$  as the data
  - Continue the process to train the next layer
- Intuitively  $\log p(\mathbf{v})$  improves as new layer is added and trained.



# Generative Pretraining a DNN

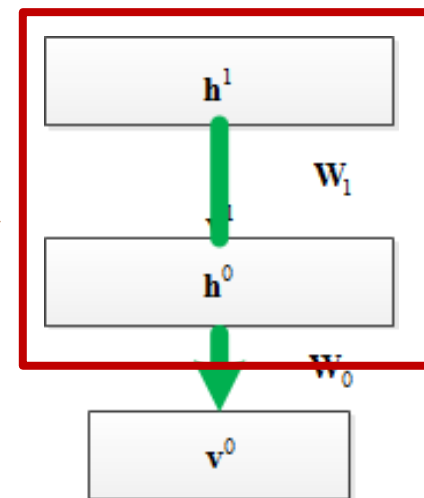
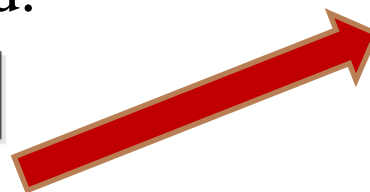
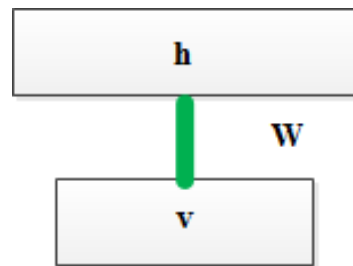
- First learn with all the weights tied
  - equivalent to learning an RBM
- Then freeze the first layer of weights and learn the remaining weights (still tied together).
  - equivalent to learning another RBM, using the aggregated conditional probability on  $\mathbf{h}_0$  as the data
  - Continue the process to train the next layer
- Intuitively  $\log p(\mathbf{v})$  improves as new layer is added and trained.



CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Generative Pretraining a DNN

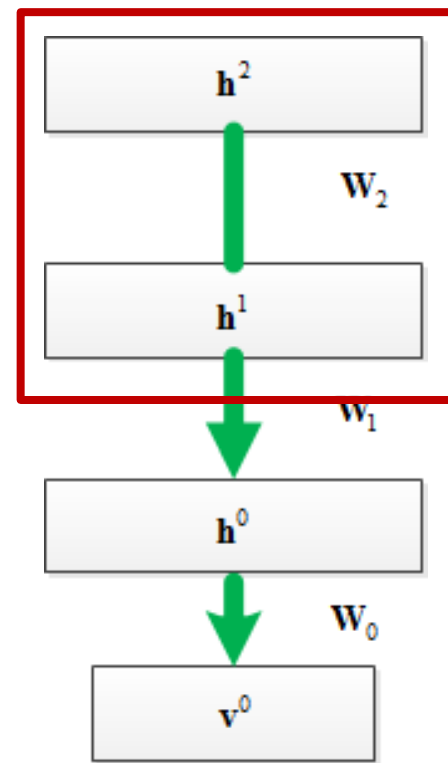
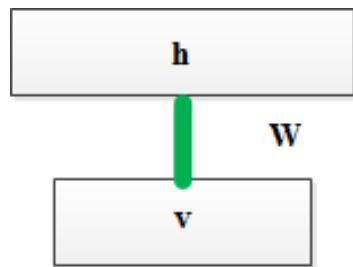
- First learn with all the weights tied
  - equivalent to learning an RBM
- Then freeze the first layer of weights and learn the remaining weights (still tied together).
  - **equivalent to learning another RBM**, using the aggregated conditional probability on  $\mathbf{h}_0$  as the data
  - Continue the process to train the next layer
- Intuitively  $\log p(\mathbf{v})$  improves as new layer is added and trained.



CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Generative Pretraining a DNN

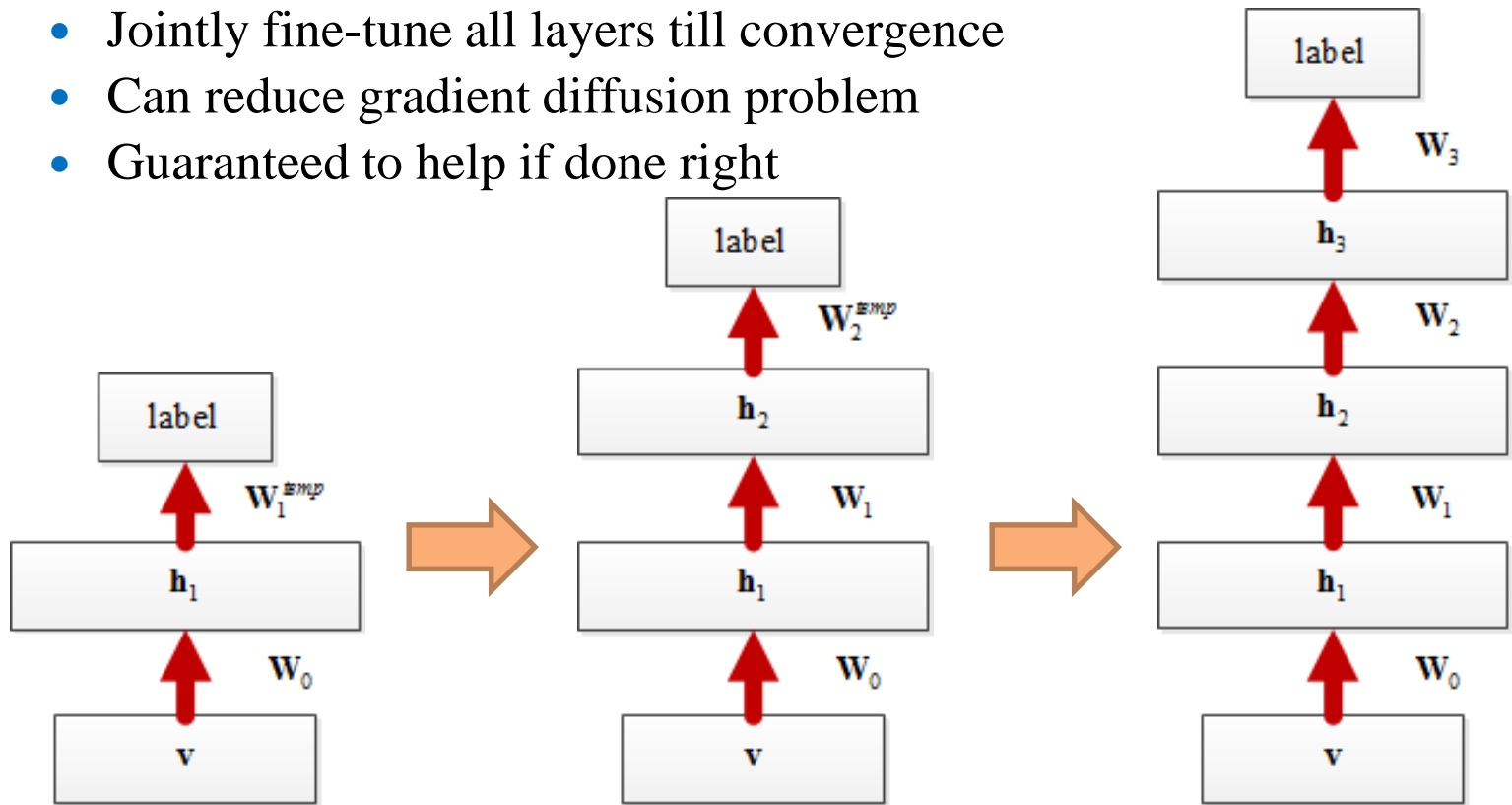
- First learn with all the weights tied
  - equivalent to learning an RBM
- Then freeze the first layer of weights and learn the remaining weights (still tied together).
  - equivalent to learning another RBM, using the aggregated conditional probability on  $\mathbf{h}_0$  as the data
  - Continue the process to train the next layer
- Intuitively  $\log p(\mathbf{v})$  improves as new layer is added and trained.



CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Discriminative Pretraining

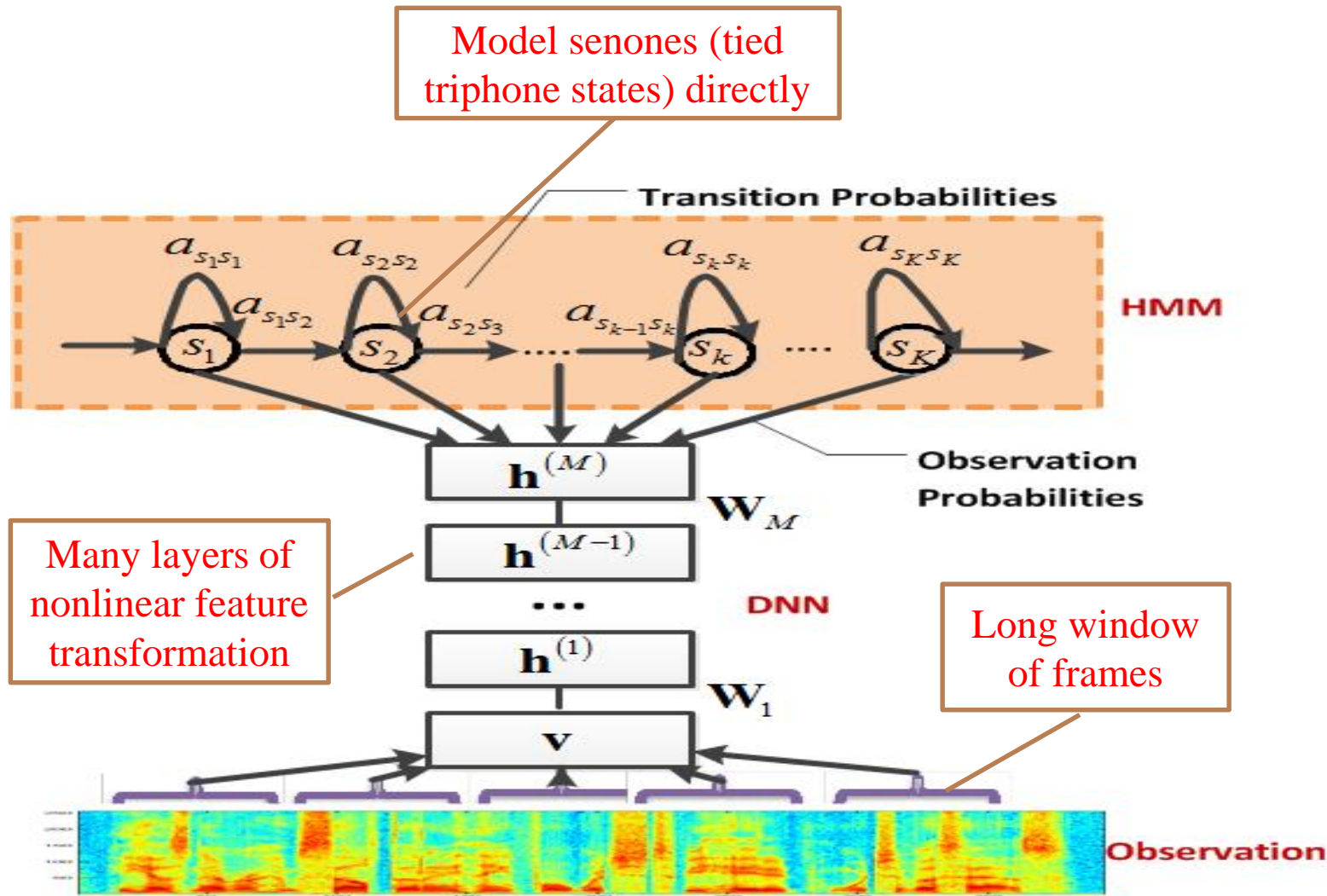
- Train a single hidden layer DNN using BP (without convergence)
- Insert a new hidden layer and train it using BP (without convergence)
- Do the same thing till the predefined number of layers is reached
- Jointly fine-tune all layers till convergence
- Can reduce gradient diffusion problem
- Guaranteed to help if done right



# CD-DNN-HMM: Three Key Components

(Dahl, Yu, Deng, Acero 2012)

CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary



# Modeling Senones is Critical

- **Table:** 24-hr Voice Search (760 24-mixture senones)

Model	monophone	senone
GMM-HMM MPE	-	36.2
DNN-HMM 1× 2K	41.7	31.9
DNN-HMM 3 × 2k	35.8	30.4

- **Table:** 309-hr SWB (9k 40-mixture senones)

Model	monophone	senone
GMM-HMM BMMI	-	23.6
DNN-HMM 7× 2K	34.9	17.1

- ML-trained CD-GMM-HMM generated alignment was used to generate senone and monophone labels for training DNNs.

# Exploiting Neighbor Frames

- **Table:** 309-hr SWB (GMM-HMM BMMI = 23.6%)

Model	1 frame	11 frames
CD-DNN-HMM 1 × 4634	26.0	22.4
CD-DNN-HMM 7 × 2k	23.2	17.1

ML-trained CD-GMM-HMM generated alignment was used to generate senone labels for training DNNs

- It seems 23.2% is only slightly better than 23.6% but note that DNN is not trained using sequence-discriminative training but GMM is.
- To exploit info in neighbor frames, GMM systems need to use fMPE, region dependent transformation, or tandem structure

# Deeper Model is More Powerful

(Seide, Li, Yu 2011, Seide, Li, Chen, Yu 2011)

- **Table:** 309-hr SWB (GMM-HMM BMMI = 23.6%)

L×N	DBN-Pretrain	1×N	DBN-Pretrain
1×2k	24.2	1×2k	24.2
2×2k	20.4	-	-
3×2k	18.4	-	-
4×2k	17.8	-	-
5×2k	17.2	1×3772	22.5
7×2k	17.1	1×4634	22.6
9×2k	17.0	-	-
9×1k	17.9	-	-
5×3k	17.0	-	-
		1×16k	22.1

CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Pretraining Helps but Not Critical

(Seide, Li, Yu 2011, Seide, Li, Chen, Yu 2011)

L×N	DBN-Pretrain	BP	LBP	Discriminative Pretrain
1×2k	24.2	24.3	24.3	24.1
2×2k	20.4	22.2	20.7	20.4
3×2k	18.4	20.0	18.9	18.6
4 ×2k	17.8	18.7	17.8	17.8
5×2k	17.2	18.2	17.4	17.1
7 ×2k	17.1	17.4	17.4	16.8
9×2k	17.0	16.9	16.9	-
9× 1k	17.9	-	-	-
5×3k	17.0	-	-	-

- Stochastic gradient alleviates the optimization problem.
- Large amount of training data alleviates the overfitting problem.
- Pretraining helps to make BP more robust

# CD-DNN-HMM Performance Summary

(Dahl, Yu, Deng, Acero 2012, Seide, Li, Yu 2011, Chen et al. 2012)

- **Table:** Voice Search SER (24 hours training)

AM	Setup	Test
GMM-HMM	MPE (760 24-mixture)	36.2%
DNN-HMM	5 layers x 2048	<b>30.1% (-17%)</b>

- **Table:** Switch Board WER (309 hours training)

AM	Setup	Hub5'00-SWB	RT03S-FSH
GMM-HMM	BMMI (9K 40-mixture)	23.6%	27.4%
DNN-HMM	7 x 2048	<b>15.8% (-33%)</b>	<b>18.5% (-33%)</b>

- **Table:** Switch Board WER (2000 hours training)

AM	Setup	Hub5'00-SWB	RT03S-FSH
GMM-HMM (A)	BMMI (18K 72-mixture)	21.7%	23.0%
GMM-HMM (B)	BMMI + fMPE	19.6%	20.5%
DNN-HMM	7 x 3076	<b>14.4% (A: -34% B: -27%)</b>	<b>15.6% (A: -32% B: -24%)</b>

# CD-DNN-HMM Real World Deploy

- **Microsoft** audio video indexing service (Knies, 2012)
  - “It’s a big deal. ... tests have demonstrated that **the algorithm provides a 10- to 20-percent relative error reduction and uses about 30 percent less processing time** than the best-of-breed speech-recognition algorithms based on so-called Gaussian Mixture Models.”
- **Google** voice search (Simonite, 2012):
  - “Google is now using these **neural networks** to recognize speech more accurately, ... **“We got between 20 and 25 percent improvement in terms of words that are wrong,”** says Vincent Vanhoucke, a leader of Google's speech-recognition efforts.
- **Microsoft** Bing voice search (Knies, 2013)
  - “With the judicious use of **DNNs**, that service has seen its **speed double** in recent weeks, and its **word-error rate has improved by 15 percent**”
- Other companies: Baidu, IBM, Nuance, Iflytech, etc.

# Part 3: Outline

---

- CD-DNN-HMM
- **Training and Decoding Speed**
- Invariant Features
- Mixed-bandwidth ASR
- Multi-lingual ASR
- Sequence Discriminative Training
- Adaptation
- Summary

# Decoding Speed

(Senior et al. 2011)

- Well within real time with careful engineering
- Setup: (1) DNN: 440:2000X5:7969 (2) single CPU (4) GPU NVIDIA Tesla C2070

Technique	Real time factor	Note
Floating-point baseline	3.89	
Floating-point SSE2	1.36	4-way parallel (16 bytes)
8-bit quantization	1.52	Hidden: unsigned char, weight: signed char
Integer SSSE3	0.51	16-way parallel
Integer SSE4	0.47	Faster 16-32 conversion
Batching	0.36	batches over tens of ms
Lazy evaluation	0.26	Assume 30% active senone
Batched lazy evaluation	0.21	Combine both

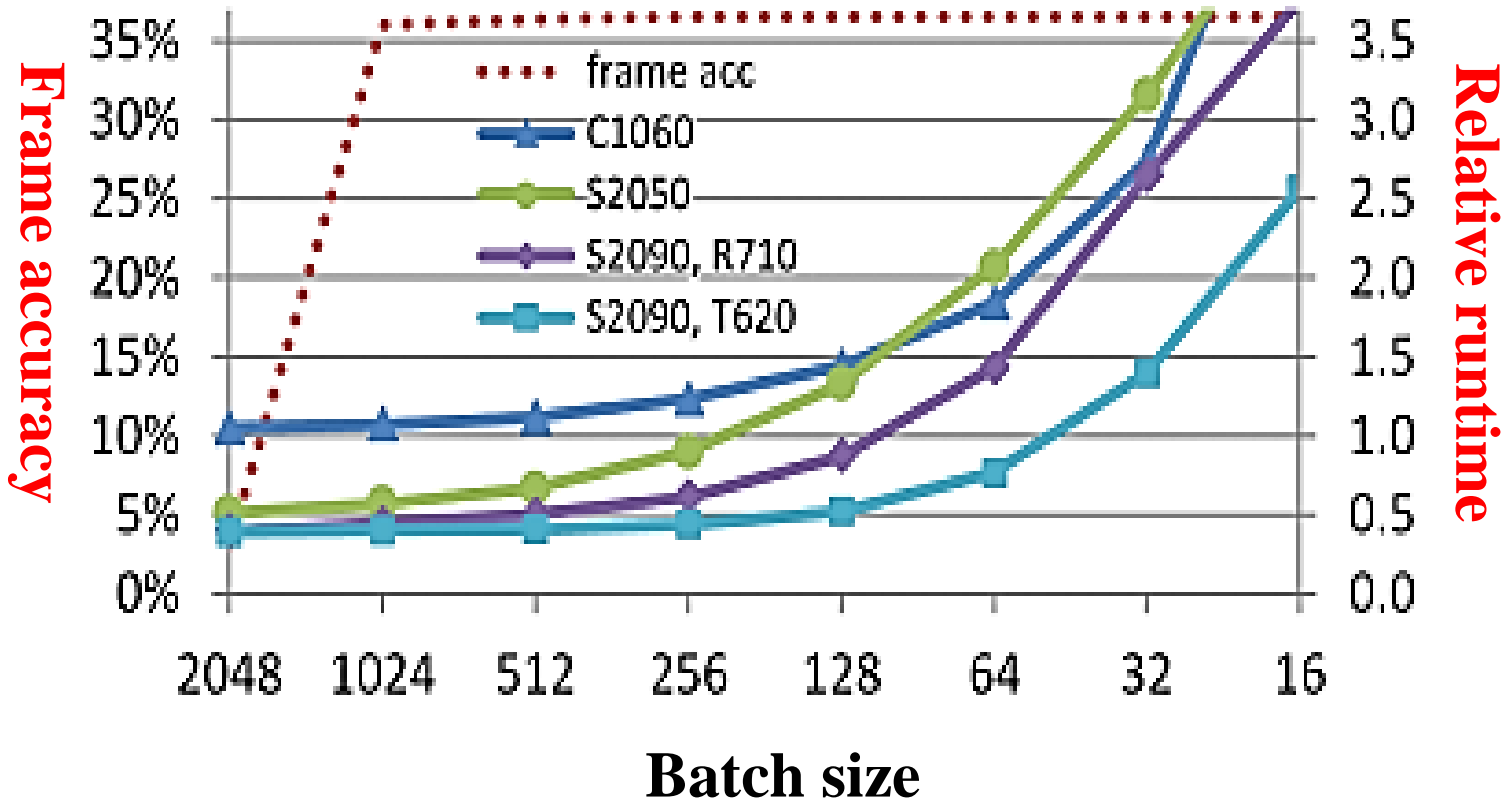
# Decoding Speed

(Xue et al. 2013)

- Observations
  - Weight sparseness: Only 20-30% weights need to be nonzero to get same performance
  - Sparse pattern is random: hard to exploit
- Low-rank factorization
  - Model the weight matrix as product of two matrices
  - SVD:  $W = U\Sigma V^T = [U\Sigma^{1/2}][\Sigma^{1/2}V^T]$
  - Keep only top singular values
  - Model size compressed to 20-30%
  - Additional three-fold speedup
  - Faster than GMM decoding

# Training (Single GPU)

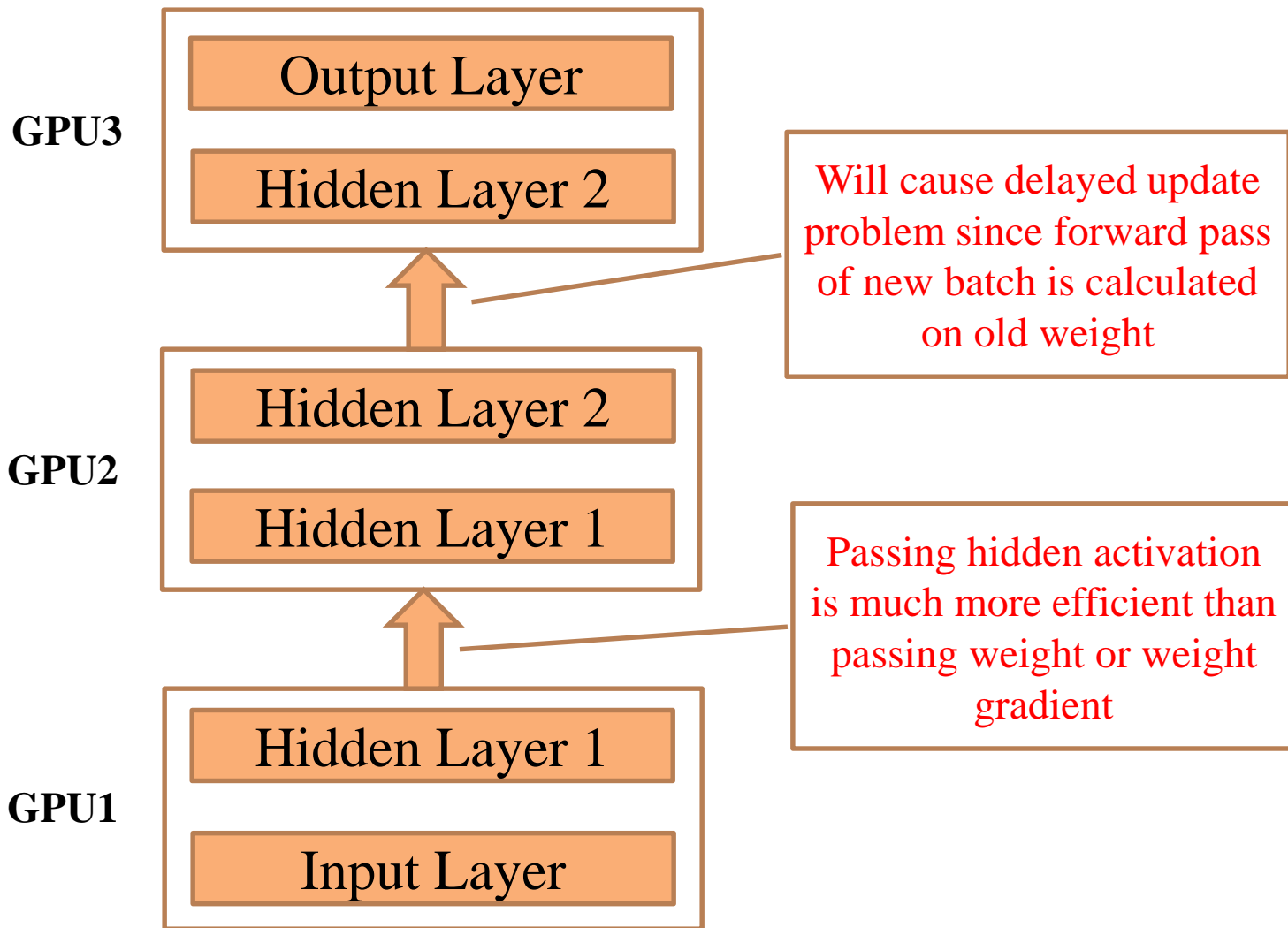
(Chen et al. 2012)



- Relative runtime for different minibatch sizes and GPU/server model types, and corresponding frame accuracy measured after seeing 12 hours of data (429:2kX7:9304).

# Training (Multi-GPU): Pipeline

(Chen et al. 2012)



CD-DNN-HMM | **Speed** | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Training (Multi-GPU): Pipeline

(Chen et al. 2012)

parallelization method	#GPU $K$	minibatch size $T$		
		256	512	1024
none (baseline)	1	68	61	59

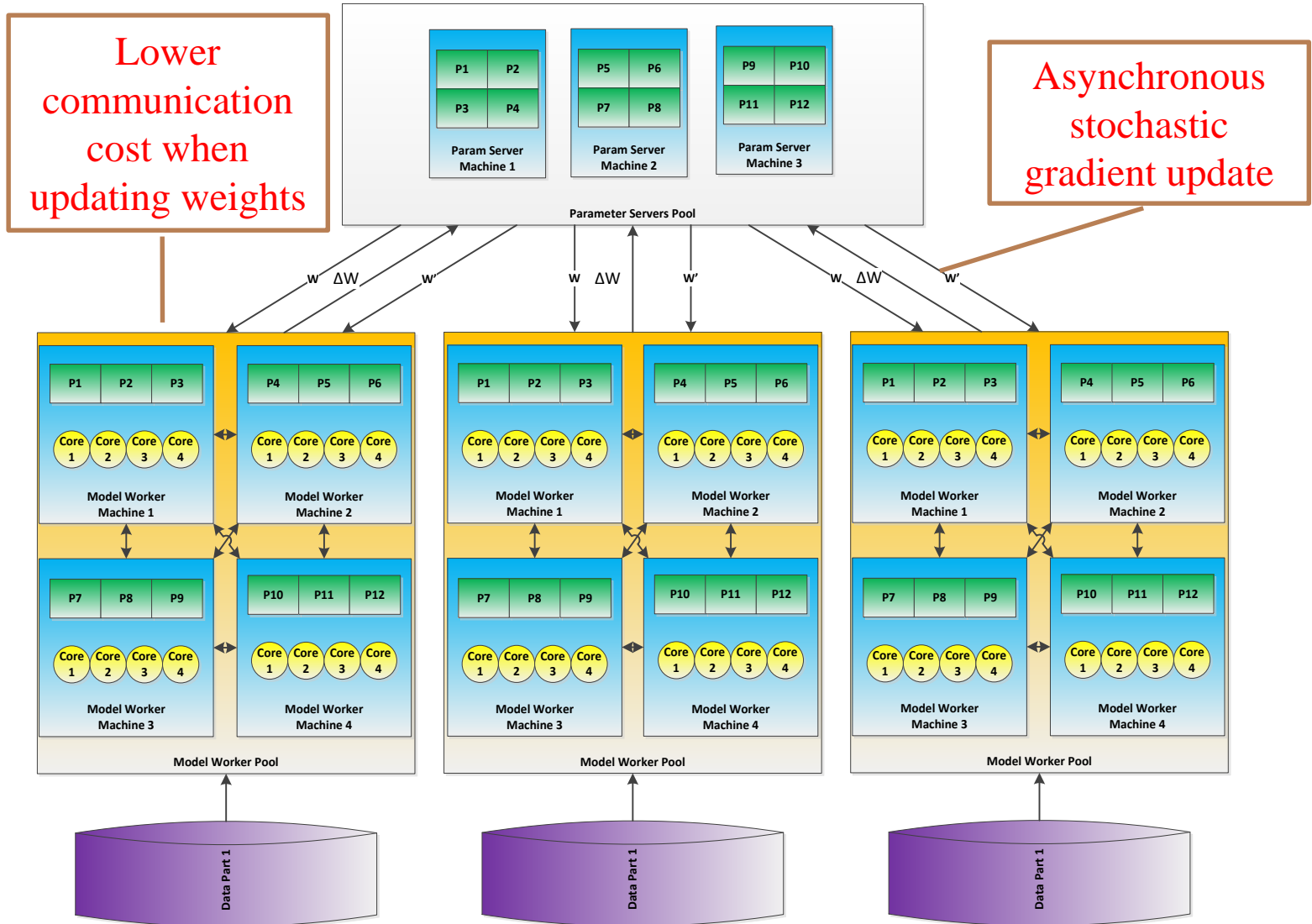
Multi-GPU with pipeline

pipeline training (0..6; 7)	2	40	34	[[33]]
vs. (0..5; 6..7)	2	36	33	31
vs. (0..2; 3..4; 5..6; 7)	4	32	29	[27]
pipeline + striped top layer	4	20	18	[[18]]

- Training runtimes in minutes per 24h of data for different parallelization configurations.  $[[\cdot]]$  denotes divergence, and  $[\cdot]$  denotes a WER loss  $> 0.1\%$  points on the Hub5 set (429:2kX7:9304).

# Training (CPU Cluster)

(Dean et al. 2012, picture courtesy of Erdinc Basci)



CD-DNN-HMM | **Speed** | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Training (CPU or GPU Cluster)

(Kingsbury et al. 2012, Martens 2010)

- Use algorithms that are effective with large batches.
  - L-BFGS (work well if you use full batch)
  - Hessian free
- Simple data parallelization would work
- Key: the communication cost is small compared to the calculation
- **Better and more scalable training algorithm is still in need.**

# Part 3: Outline

---

- CD-DNN-HMM
- Training and Decoding Speed
- **Invariant Features**
- Mixed-bandwidth ASR
- Multi-lingual ASR
- Sequence Discriminative Training
- Adaptation
- Summary

# What Makes ASR Difficult?

## Variability, Variability, Variability

### Speaker

- Accents
- Dialect
- Style
- Emotion
- Coarticulation
- Reduction
- Pronunciation
- Hesitation
- ...

### Environment

- Noise
- Side talk
- Reverberation
- ...

### Device

- Head phone
- Land phone
- Speaker phone
- Cell phone
- ...

Interactions between these factors are complicated and nonlinear

# DNN Is Powerful and Efficient

- The desirable model should be **powerful** and **efficient** to represent complex structures
  - DNN can model any mapping (**powerful**): universal approximator -> same as shallow model
  - DNN is **efficient** in representation: need fewer computational units for the same function by sharing lower-layer results -> better than shallow models
- **DNN learns invariant and discriminative features**

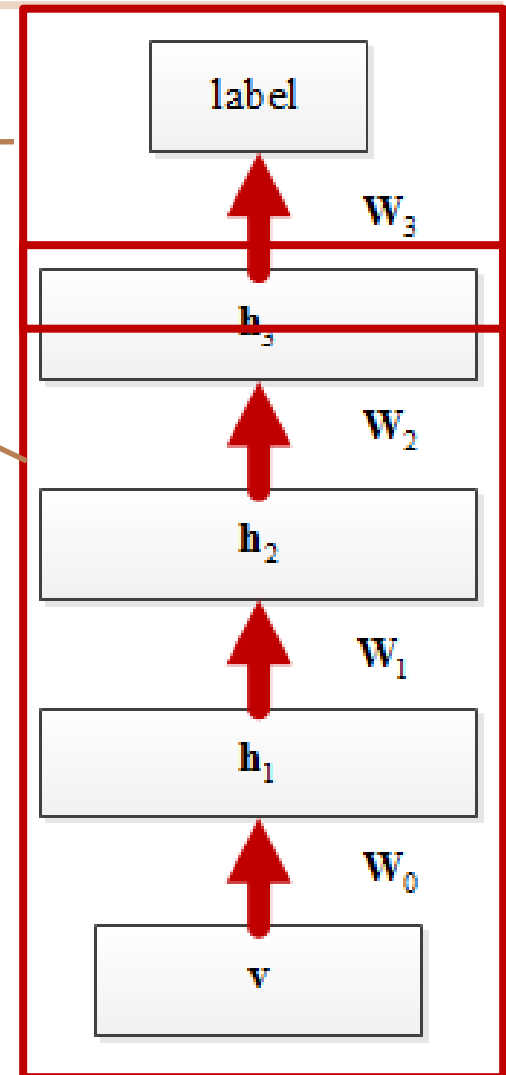
CD-DNN-HMM | Speed | **Invariant Features** | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Research DNN Learns Invariant and Discriminative Features

Log-linear classifier

Many layers of nonlinear feature transformation

- **Joint** feature learning and classifier design
  - Bottleneck or tandem feature does not have this property
- Many simple non-linearities = One complicated non-linearity
- Features at **higher layers are more invariant and discriminative** than those at lower layers



# Higher Layer Features More Invariant

(yu et al. 2013)

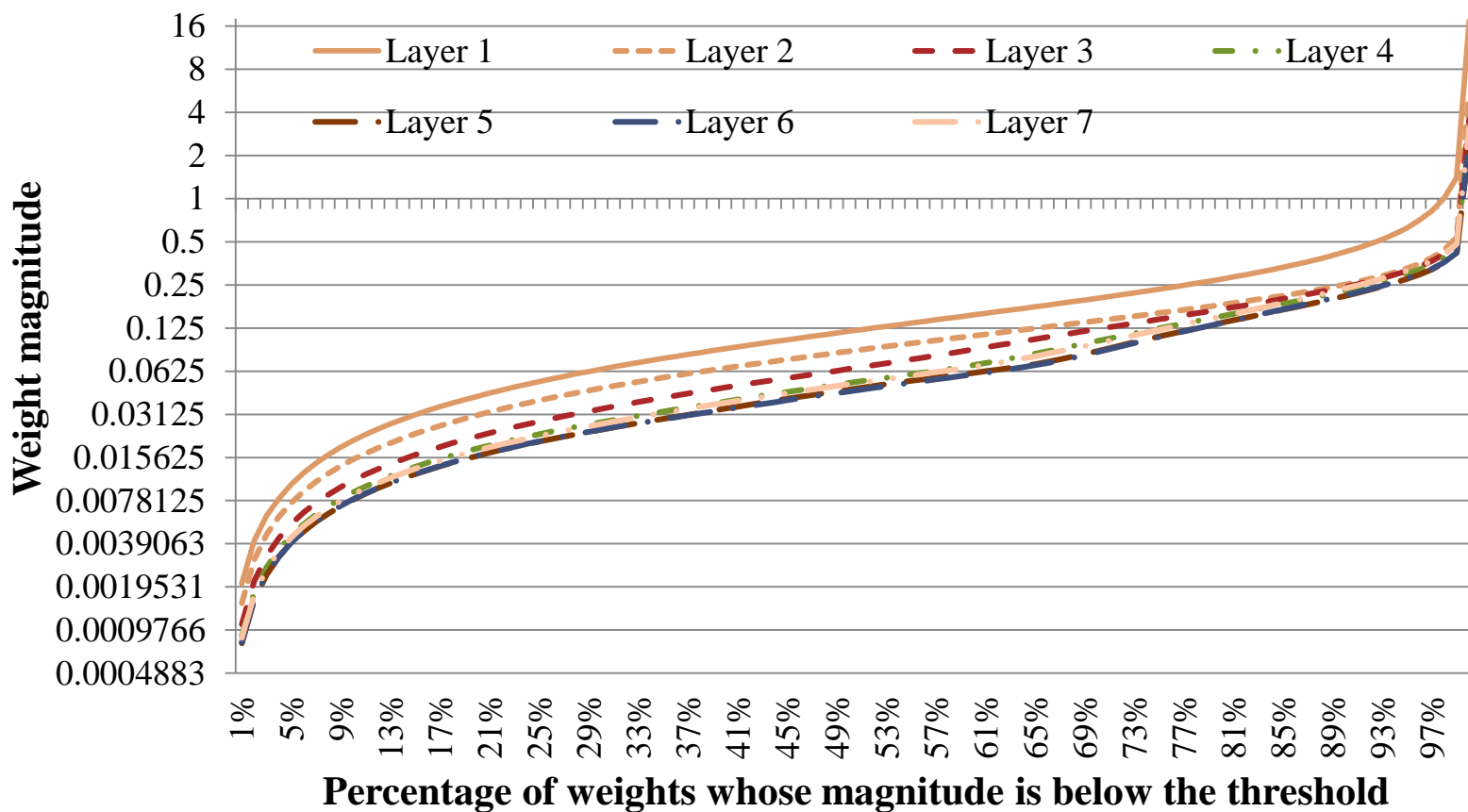
$$\|\delta^{l+1}\| = \|\sigma(z^l(v^l + \delta^l)) - \sigma(z^l(v^l))\| \cong \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) \left((w^l)^T \delta^l\right) \right\| \leq \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) (w^l)^T \right\| \|\delta^l\|$$

CD-DNN-HMM | Speed | **Invariant Features** | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Higher Layer Features More Invariant

(yu et al. 2013)

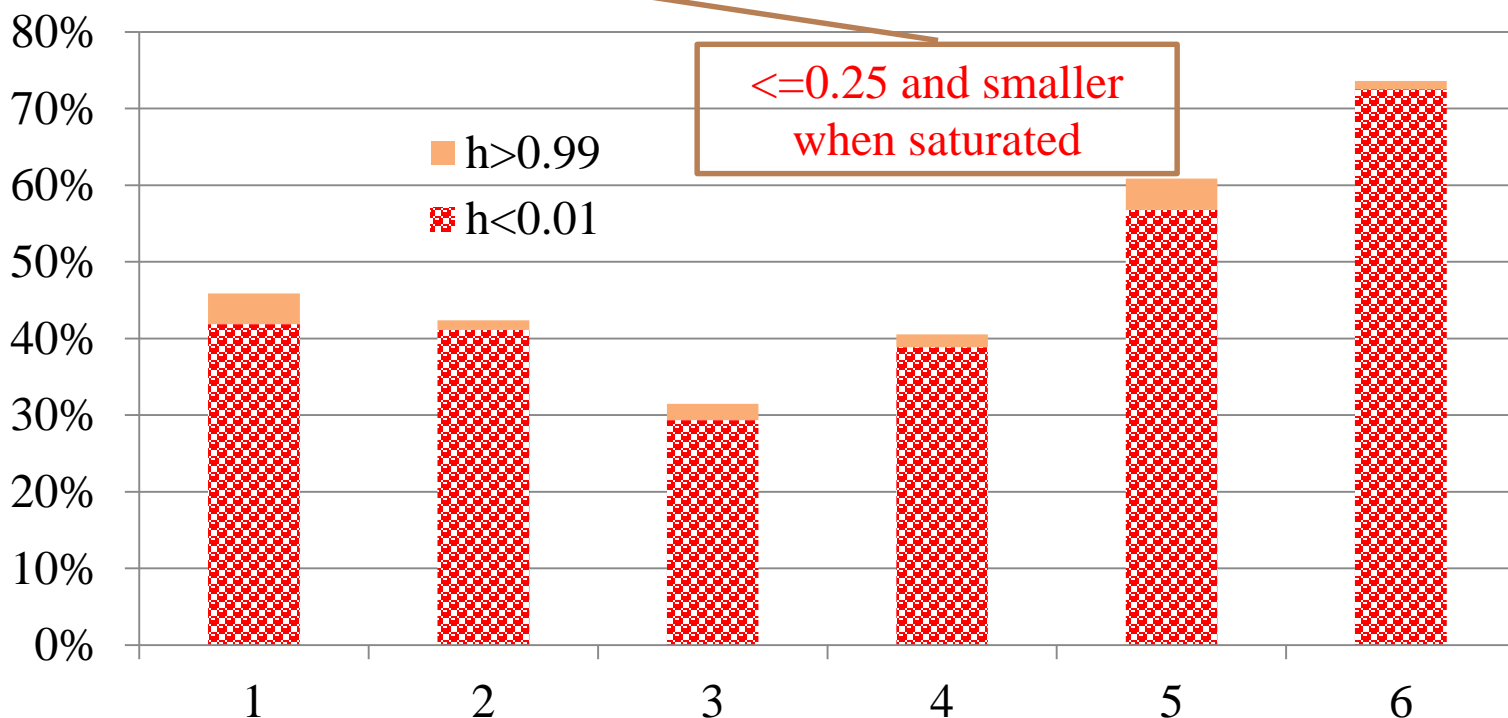
$$\|\delta^{l+1}\| = \|\sigma(z^l(v^l + \delta^l)) - \sigma(z^l(v^l))\| \cong \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) \left((w^l)^T \delta^l\right) \right\| \leq \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) (w^l)^T \right\| \|\delta^l\|$$



# Higher Layer Features More Invariant

(yu et al. 2013)

$$\|\delta^{l+1}\| = \|\sigma(z^l(v^l + \delta^l)) - \sigma(z^l(v^l))\| \cong \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) \left((w^l)^T \delta^l\right) \right\| \leq \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) (w^l)^T \right\| \|\delta^l\|$$

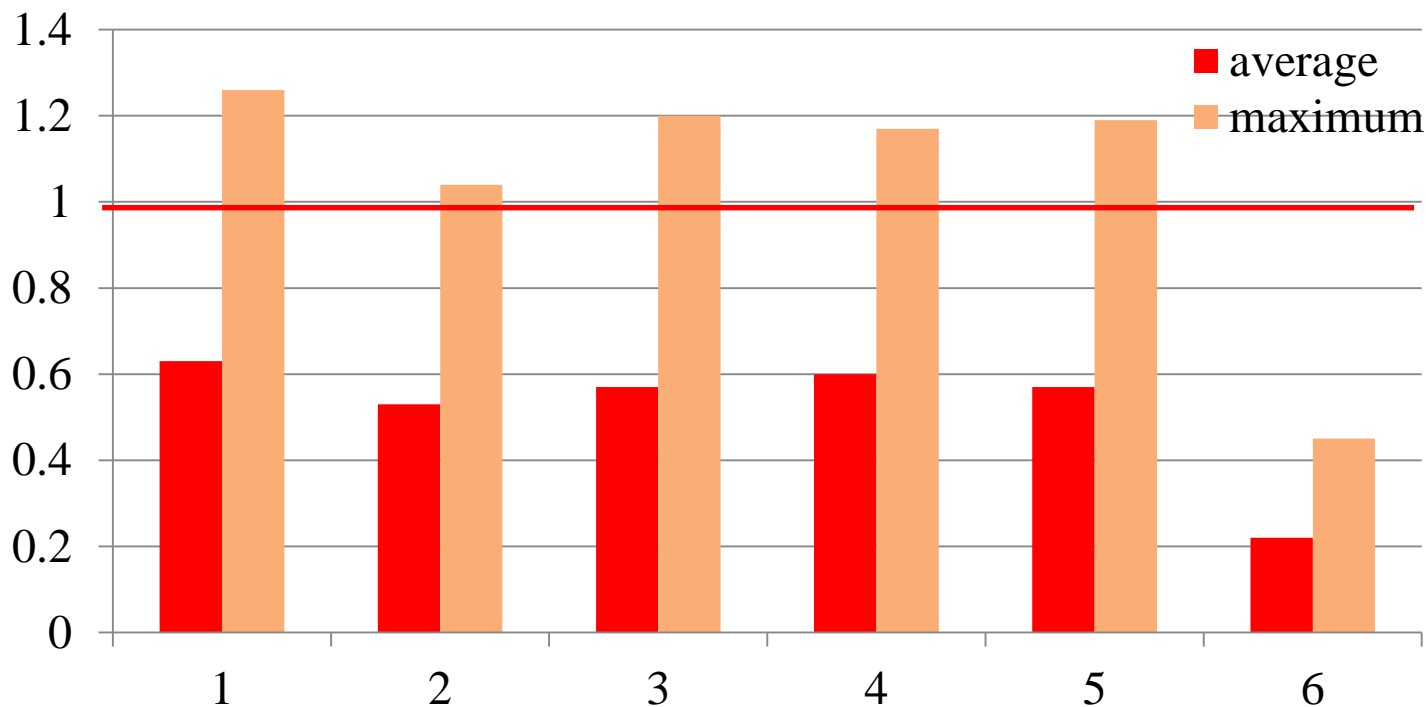


- Percentage of saturated hidden units.
- H<0.01 are inactive neurons. Higher layers are more sparse

# Higher Layer Features More Invariant

(yu et al. 2013)

$$\|\delta^{l+1}\| = \|\sigma(z^l(v^l + \delta^l)) - \sigma(z^l(v^l))\| \cong \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) \left((w^l)^T \delta^l\right) \right\| \leq \left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) (w^l)^T \right\| \|\delta^l\|$$



$$\left\| \text{diag}\left(\sigma'(z^l(v^l(t)))\right) (w^l)^T \right\|$$

If the norm <1 the variation shrinks one layer higher

# Noise Robustness

(Seltzer, Yu, Wang 2013)

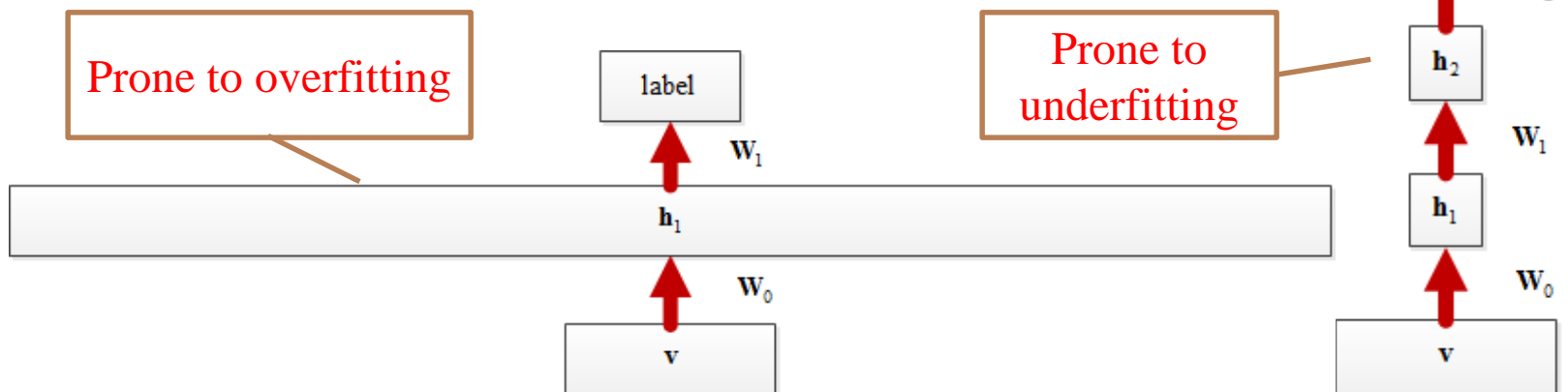
- DNN converts input features into more invariant and discriminative features
- Robust to environment and speaker variations
- Aurora 4 16kHz medium vocabulary noise robustness task
  - Training: 7137 utterances from 83 speakers
  - Test: 330 utterances from 8 speakers

**Table:** WER (%) Comparison on Aurora4 (16k Hz) Dataset.

Setup	Set A	Set B	Set C	Set D	Avg
GMM-HMM (Baseline)	12.5	18.3	20.5	31.9	23.9
GMM-HMM (MPE + VAT)	7.2	12.8	11.5	19.7	15.3
GMM-HMM + Structured SVM	7.4	12.6	10.7	19.0	14.8
GMM-HMM (VAT-Joint)	5.6	11.0	8.8	17.8	13.4
CD-DNN-HMM (2kx7)	5.6	8.8	8.9	20.0	13.4
CD-DNN-HMM+NAT+dropout	5.4	8.3	7.6	18.5	12.4

# Balance Overfitting and Underfitting

- Achieved by adjusting width and depth -> shallow model is lack of depth adjustment
- DNN adds constrains to the space of transformations – less likely to overfit
- Larger variability -> wider layers
- Small training set -> narrower layers
- A good system is deep and wide.



# Part 3: Outline

---

- CD-DNN-HMM
- Training and Decoding Speed
- Invariant Features
- **Mixed-bandwidth ASR**
- Multi-lingual ASR
- Sequence Discriminative Training
- Adaptation
- Summary

# Flexible in Using Features

(Mohamed et al. 2012, Li et al. 2012)

- Information and features that cannot be effectively exploited within the GMM framework can now be exploited

**Table:** Comparison of different input features for DNN. All the input features are mean-normalized and with dynamic features. Relative WER reduction in parentheses.

Setup	WER (%)
CD-GMM-HMM (MFCC, fMPE+BMMI)	34.66 (baseline)
CD-DNN-HMM (MFCC)	31.63 (-8.7%)
CD-DNN-HMM (24 <b>log filter-banks</b> )	30.11 (-13.1%)
CD-DNN-HMM (29 log filter-banks)	30.11 (-13.1%)
CD-DNN-HMM (40 log filter-banks)	29.86 (-13.8%)
CD-DNN-HMM (256 log FFT bins)	32.26 (-6.9%)

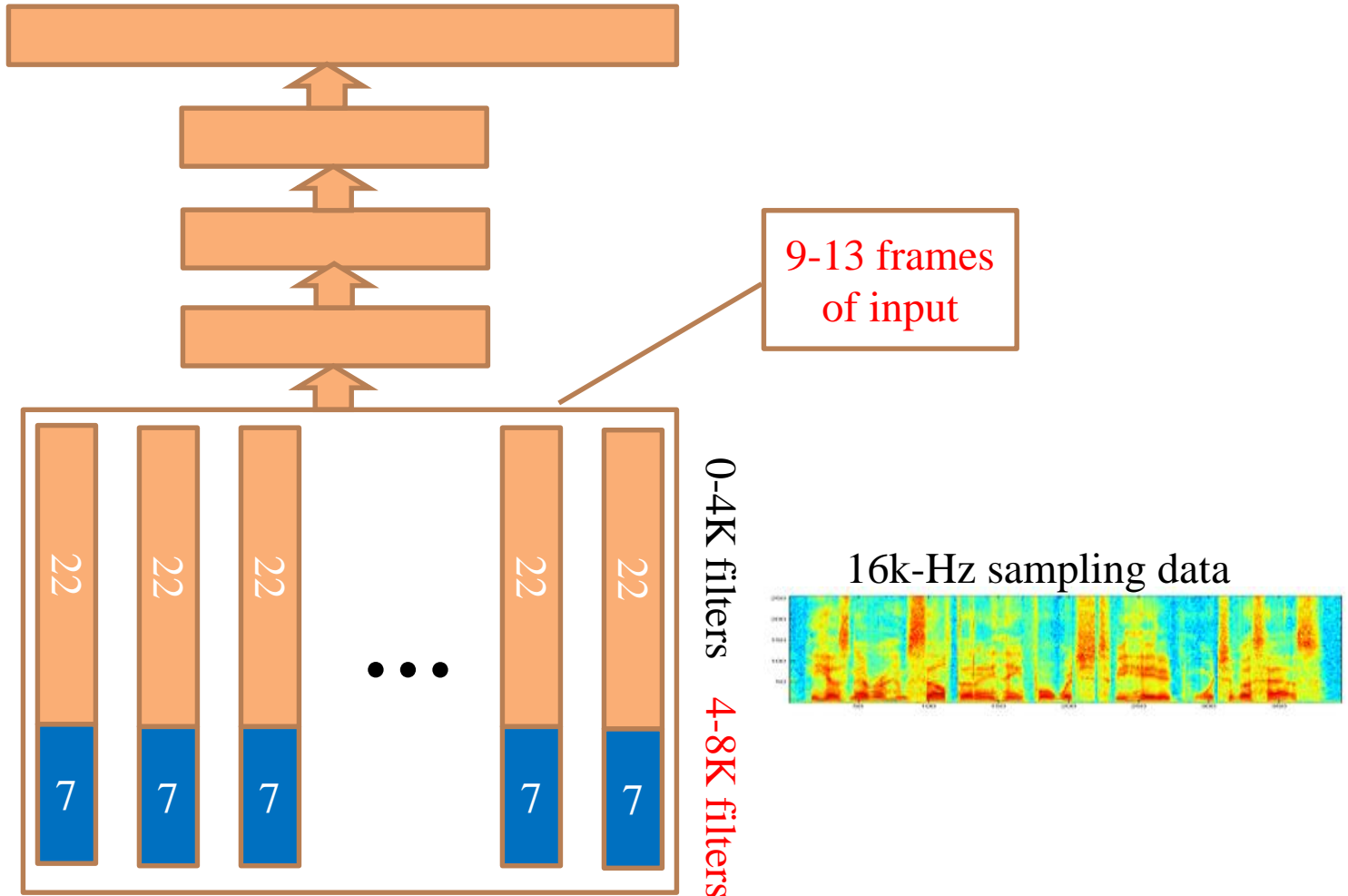
Training set: VS-1 72 hours of audio.

Test set: VS-T (26757 words in 9562 utterances).

Both the training and test sets were collected at 16-kHz sampling rate.

# Mixed Bandwidth ASR

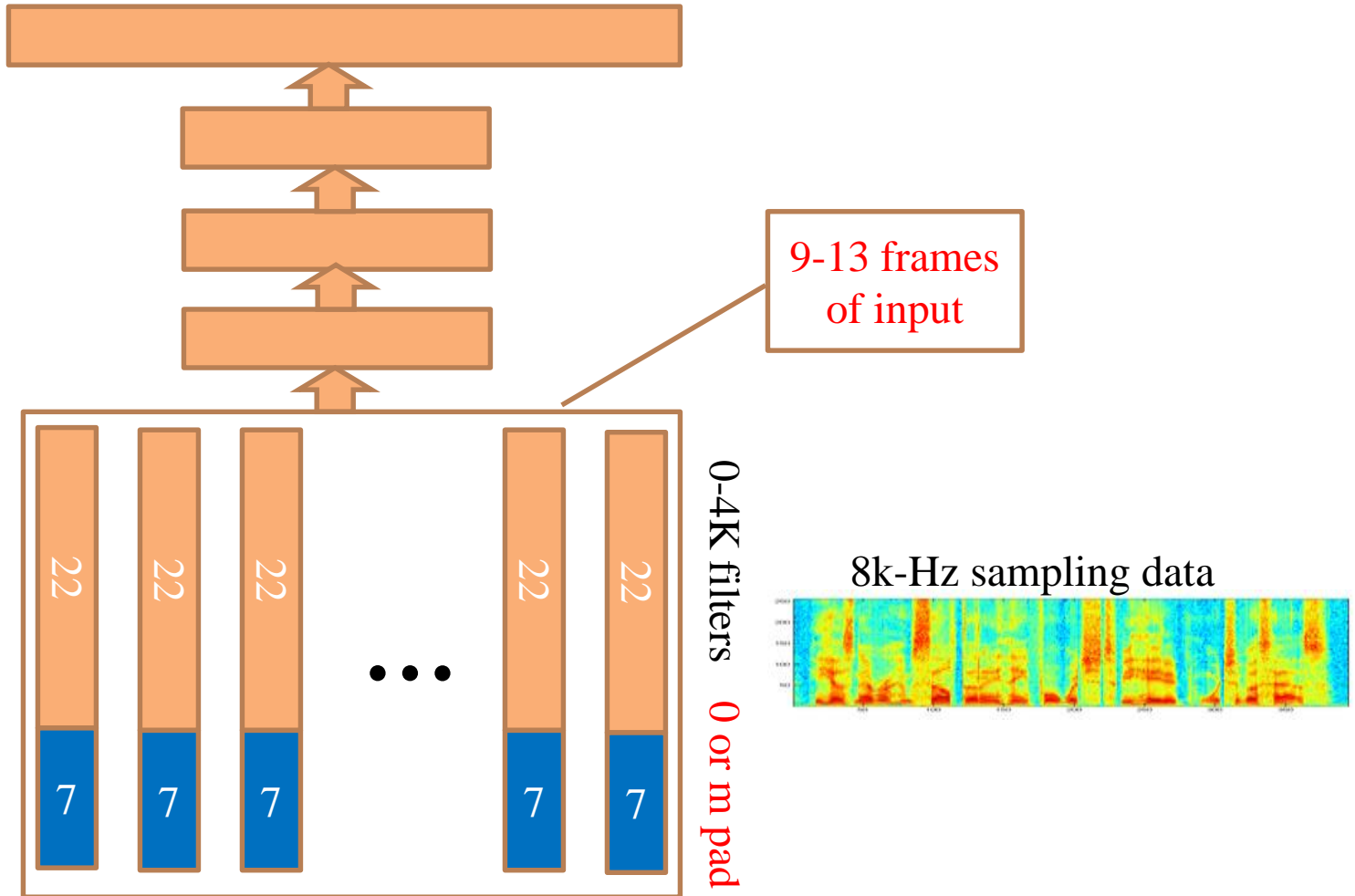
(J. Li et. al 2012)



**Figure:** DNN training/testing with 16-kHz and 8-kHz sampling data

# Mixed Bandwidth ASR

(J. Li et. al 2012)



**Figure:** DNN training/testing with 16-kHz and 8-kHz sampling data

# Mixed Bandwidth ASR

(J. Li et. Al 2012)

**Table:** DNN performance on wideband and narrowband test sets using mixed-bandwidth training data.

Training Data	WER (16-kHz VS-T)	WER (8-kHz VS-T)
16-kHz VS-1 (B1)	<b>29.96</b>	71.23
8-kHz VS-1 + 8-kHz VS-2 (B2)	-	<b>28.98</b>
16-kHz VS-1 + 8-kHz VS-2 (ZP)	<b>28.27</b>	29.33
16-kHz VS-1 + 8-kHz VS-2 (MP)	28.36	29.37
16-kHz VS-1 + 16-kHz VS-2 (UB)	<b>27.47</b>	53.51

B1: baseline 1

B2: baseline 2

ZP: zero padding

MP: mean padding

UB: upper bound

Mixed-bandwidth: recover 2/3 of (UB-B1) and 1/2 of (UB-B2)

# Mixed Bandwidth ASR

(J. Li et. al 2012)

**Table:** The Euclidean distance (ED) for the output vectors at each hidden layer (L1-L7) and the KL-divergence (in nats) for the posterior vectors at the top layer between 8-kHz and 16-kHz input features

	16-kHz DNN (UB)		Data-mix DNN (ZP)	
Layer	Mean (ED)	Variance (ED)	Mean (ED)	Variance (ED)
L1	13.28	3.90	7.32	3.62
L2	10.38	2.47	5.39	1.28
L3	8.04	1.77	4.49	1.27
L4	8.53	2.33	4.74	1.85
L5	9.01	2.96	5.39	2.30
L6	8.46	2.60	4.75	1.57
L7	5.27	1.85	3.12	0.93
Layer	Mean (KL)		Mean (KL)	
Top layer	2.03		0.22	

CD-DNN-HMM | Speed | Invariant Features | **Mixed-bandwidth** | Multi-lingual | Sequence Train | Adaptation | Summary

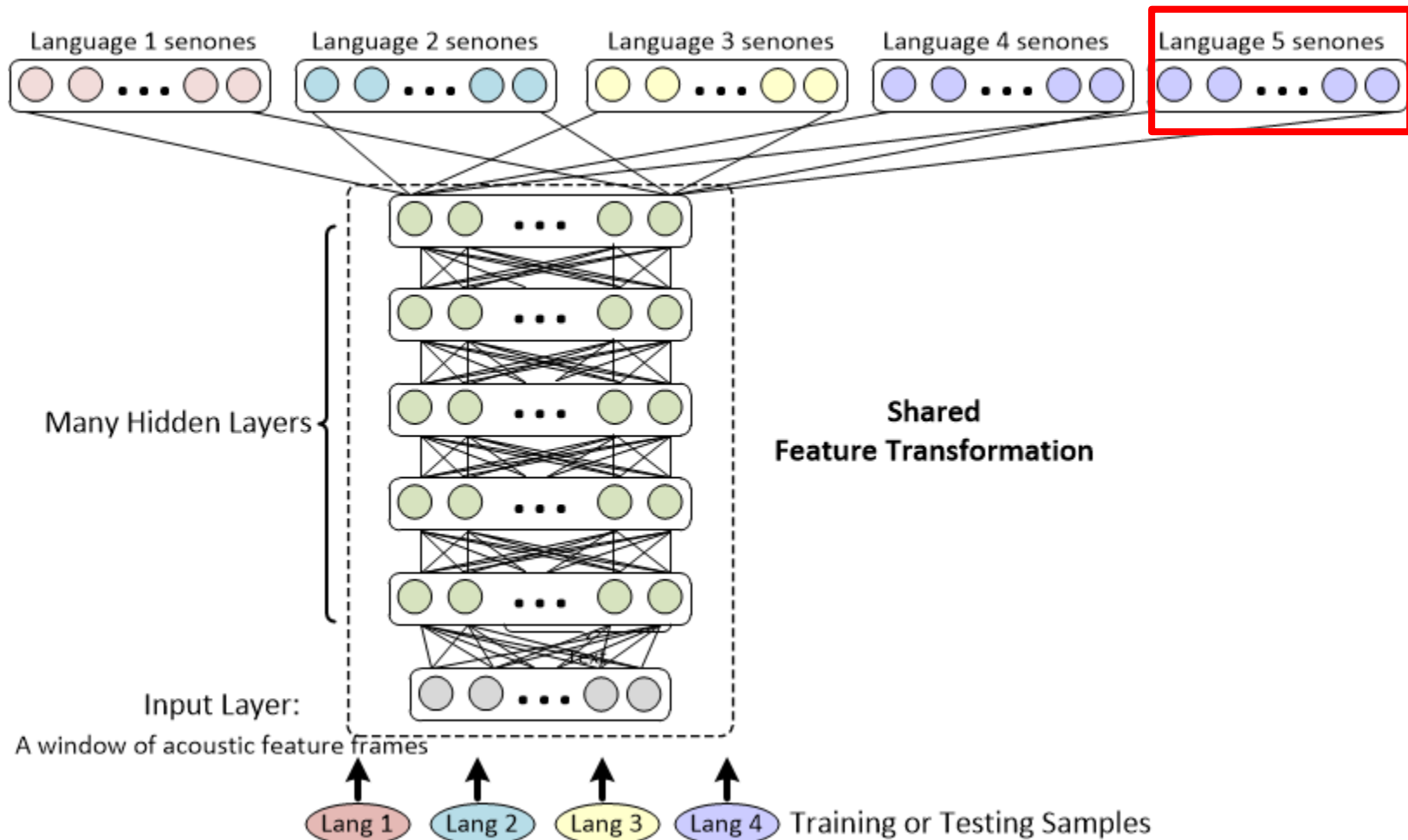
# Part 3: Outline

---

- CD-DNN-HMM
- Training and Decoding Speed
- Invariant Features
- Mixed-bandwidth ASR
- **Multi-lingual ASR**
- Sequence Discriminative Training
- Adaptation
- Summary

# Multilingual DNN

(Huang et. al 2013)



CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | **Multi-lingual** | Sequence Train | Adaptation | Summary

# Sum Is Greater Than One

**Table:** Compare Monolingual DNN and Shared-Hidden-Layer Multilingual DNN in WER (%)

	FRA	DEU	ESP	ITA
<b>Test Set Size (Words)</b>	40k	37k	18k	31k
<b>Monolingual DNN (%)</b>	28.1	24.0	30.6	24.3
<b>SHL-MDNN (%)</b>	27.1	22.7	29.4	23.5
<b>Relative WER Reduction (%)</b>	3.5	5.4	4.0	3.4

CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | **Multi-lingual** | Sequence Train | Adaptation | Summary

# Hidden Layers Are Transferable

**Table:** Compare ENU WER with and without Using Hidden Layers (HLs) Transferred from the FRA DNN.

	WER (%)
<b>Baseline (9-hr ENU)</b>	30.9
<b>FRA HLs + Train All Layers</b>	30.6
<b>FRA HLs + Train Softmax Layer</b>	27.3
<b>SHL-MDNN + Train Softmax Layer</b>	25.3

**Table:** Compare the Effect of Target Language Training Set Size in WER (%) when SHLs Are Transferred from the SHL-MDNN

ENU training data (#. Hours)	3	9	36
<b>Baseline DNN (no Transfer)</b>	38.9	30.9	23.0
<b>SHL-MDNN + Train Softmax Layer</b>	28.0	25.3	22.4
<b>SHL-MDNN + Train All Layers</b>	33.4	28.9	21.6
<b>Best Case Relative WER Reduction (%)</b>	28.2	18.0	6.3

CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Hidden Layers Are Transferable

**Table:** Effectiveness of Cross-Lingual Model Transfer on CHN Measured in CER Reduction (%).

CHN Training Set (Hrs)	3	9	36	139
<b>Baseline - CHN only</b>	45.1	40.3	31.7	29.0
<b>SHL-MDNN Model Transfer</b>	35.6	33.9	28.4	26.6
<b>Relative CER Reduction</b>	21.0	15.7	10.3	8.1

**But only if trained with labeled data**

**Table:** Compare Features Learned from Multilingual Data with and without Using Label Information on ENU Data

	WER (%)
<b>Baseline (9-hr ENU)</b>	30.9
<b>SHL-MDNN + Train Softmax Layers (no label)</b>	38.7
<b>SHL-MDNN + Train All Layers (no label)</b>	30.2
<b>SHL-MDNN + Train Softmax Layers (use label)</b>	25.3

CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | **Multi-lingual** | Sequence Train | Adaptation | Summary

# Part 3: Outline

---

- CD-DNN-HMM
- Training and Decoding Speed
- Invariant Features
- Mixed-bandwidth ASR
- Multi-lingual ASR
- **Sequence Discriminative Training**
- Adaptation
- Summary

# Sequence Discriminative Training

- Sequence-discriminative training can achieve additional gain similar to MPE and BMMI on GMM
- State-level minimum Bayes risk (sMBR), MMI and BMMI.
- **Table:** Broad cast news Dev-04f (Sainath et. al 2011)

Training Criterion	1504 senones
Frame-level Cross Entropy	18.5%
Sequence-level Criterion (sMBR)	17.0%

- **Table:** SWB (309-hr) (Kingsbury et al. 2012)

AM	Setup	Hub5'00-SWB	RT03S-FSH
SI GMM-HMM	BMMI+fMPE	18.9%	22.6%
SI DNN-HMM	7 x 2048 (frame CE)	16.1%	18.9%
SA GMM-HMM	BMMI+fMPE	15.1%	17.6%
SI DNN-HMM	7 x 2048 (sMBR)	13.3%	16.4%

# Sequence Discriminative Training

- To make it work is not easy
- Main problem: Overfitting
  - Lattice is sparse: 300 out of 9000 senones are seen in the lattice
  - Silence model can eat other models: deletion increased
  - Depends on information (e.g., LM, surrounding labels) that may be highly mismatched
- Solution:
  - F-smoothing: Hybrid training criterion: sequence-level + frame-level
  - Frame-dropout: remove frames where the reference hypothesis is missing from the lattice.
- **Table:** SWB (309-hr 7x2048 0.9s+0.1f) (Su et al. 2013)

AM	Setup	Hub5'00-SWB	RT03S-FSH
SI DNN-HMM	frame CE	16.2%	19.6%
SI DNN-HMM	MMI w/ F-smoothing)	13.5%	17.1%

# Part 3: Outline

---

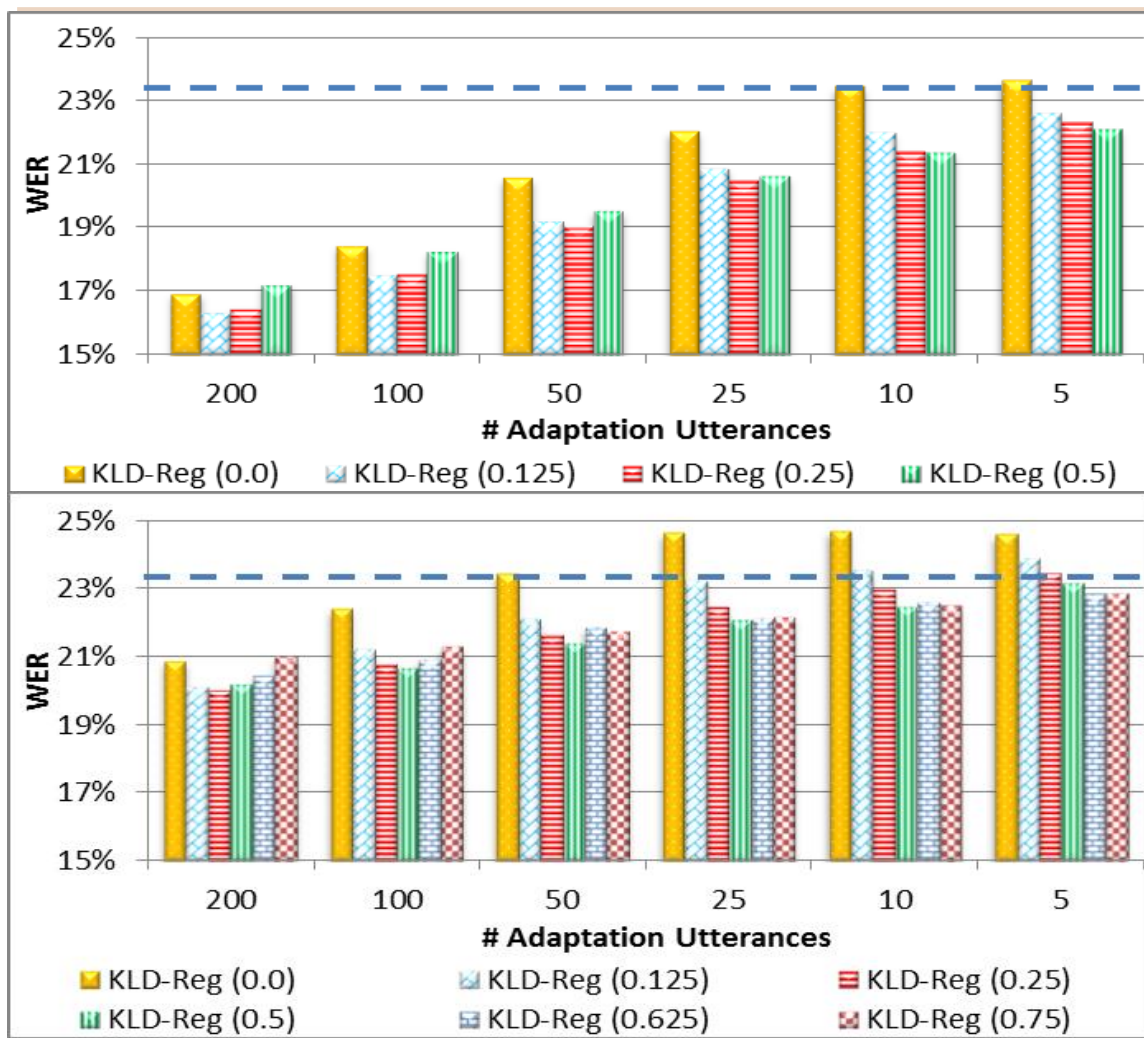
- CD-DNN-HMM
- Training and Decoding Speed
- Invariant Features
- Mixed-bandwidth ASR
- Multi-lingual ASR
- Sequence Discriminative Training
- **Adaptation**
- Summary

# KL-Regularized Adaptation

(Yu et. al 2013)

- Huge number of parameters vs. very small adaptation set
- Trick: conservative adaptation
  - the posterior senone distribution estimated from the adapted model should not deviate too far away from that estimated using the unadapted model
  - Use KL-divergence regularization on the DNN output
  - Equivalent to change the target distribution to  $\hat{p}(y|x_t) \triangleq (1 - \rho)\tilde{p}(y|x_t) + \rho p^{SI}(y|x_t)$ .

# KL-Regularized Adaptation



*Supervised Adaptation*

*Unsupervised Adaptation*

**Figure:** WERs on the SMD dataset using KLD-Reg adaptation for different regularization weights  $\rho$  (numbers in parentheses). The dashed line is the SI DNN baseline.

# KL-Regularized Adaptation

**Table.** WER and Relative WER Reduction (in parentheses) on Lecture Transcription Task:

- SI DNN: trained with 2000hr SWB
- Adaptation set: 6 lectures or 3.8 hrs
- Dev set: 1 lecture 5612 words
- Test set: 1 lecture 8481 words

	SI DNN	Supervised	Unsupervised	SD DNN
<b>Dev Set</b>	16.0%	14.3% (10.6%)	14.9% (6.9%)	15.0%
<b>Test Set</b>	20.9%	19.1% (8.6%)	19.4% (7.2%)	20.2%
<b>Chan Mismatch</b>	14.2%	16.0%	14.6%	

- **Problems:**
  - Cannot factorize impact from speaker and channel
  - Footprint each speaker is too large (there are many solutions to this)

# Part 3: Outline

---

- CD-DNN-HMM
- Training and Decoding Speed
- Invariant Features
- Mixed-bandwidth ASR
- Multi-lingual ASR
- Sequence Training Criterion
- Adaptation
- **Summary**

CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | **Summary**

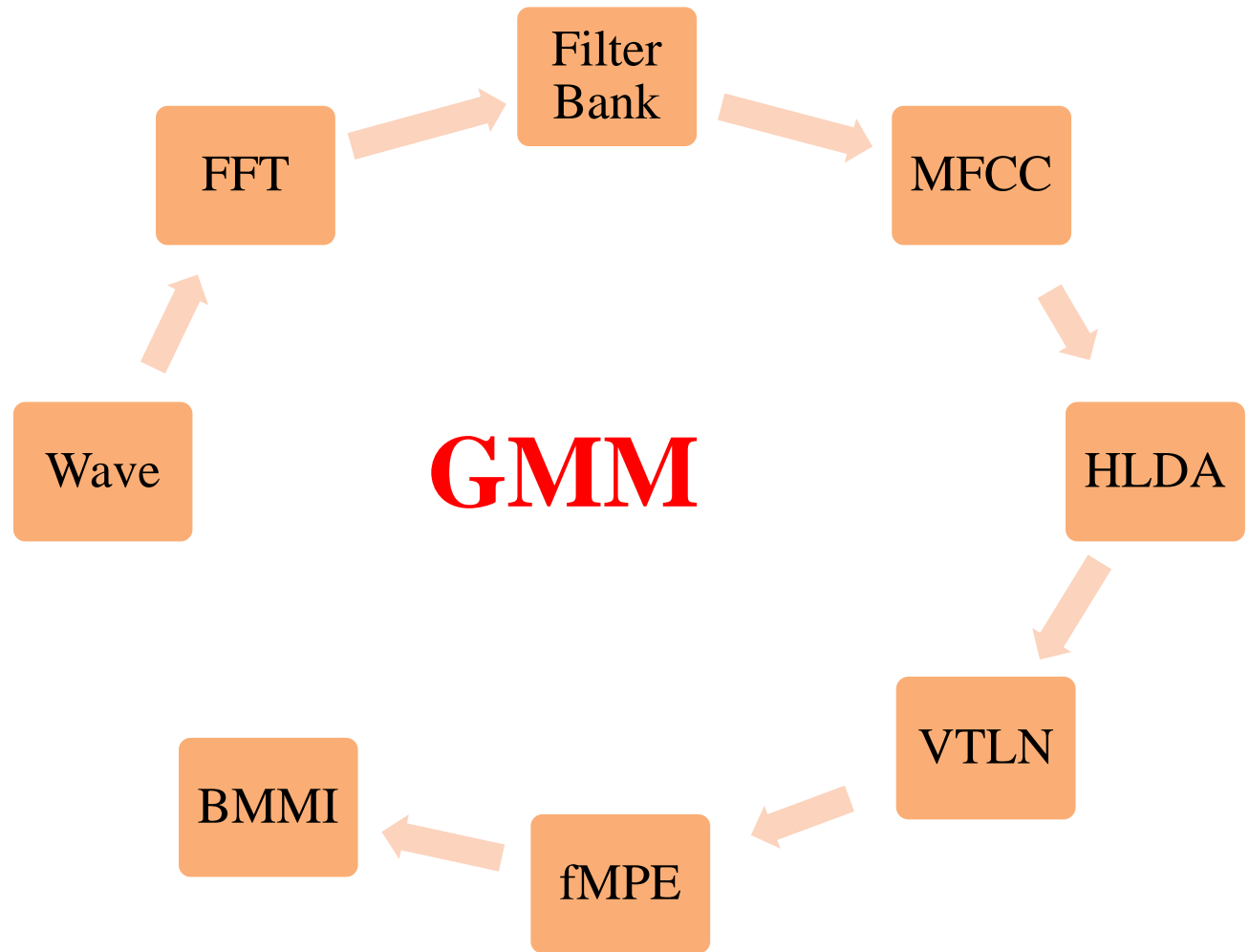
# Summary

- DNN already outperforms GMM in many tasks
  - Deep neural network is more powerful than the shallow models including GMMs
  - Features learned by DNNs are more invariant and selective
  - DNNs can exploit more info and features difficult to exploit in the GMM framework
- Many speech groups (Microsoft, Google, IBM) are adopting it.
- Commercial deployment of DNN systems is practical now
  - Many once considered obstacles for adopting DNNs have been removed
  - Already commercially deployed by Microsoft and Google

# Other Advances

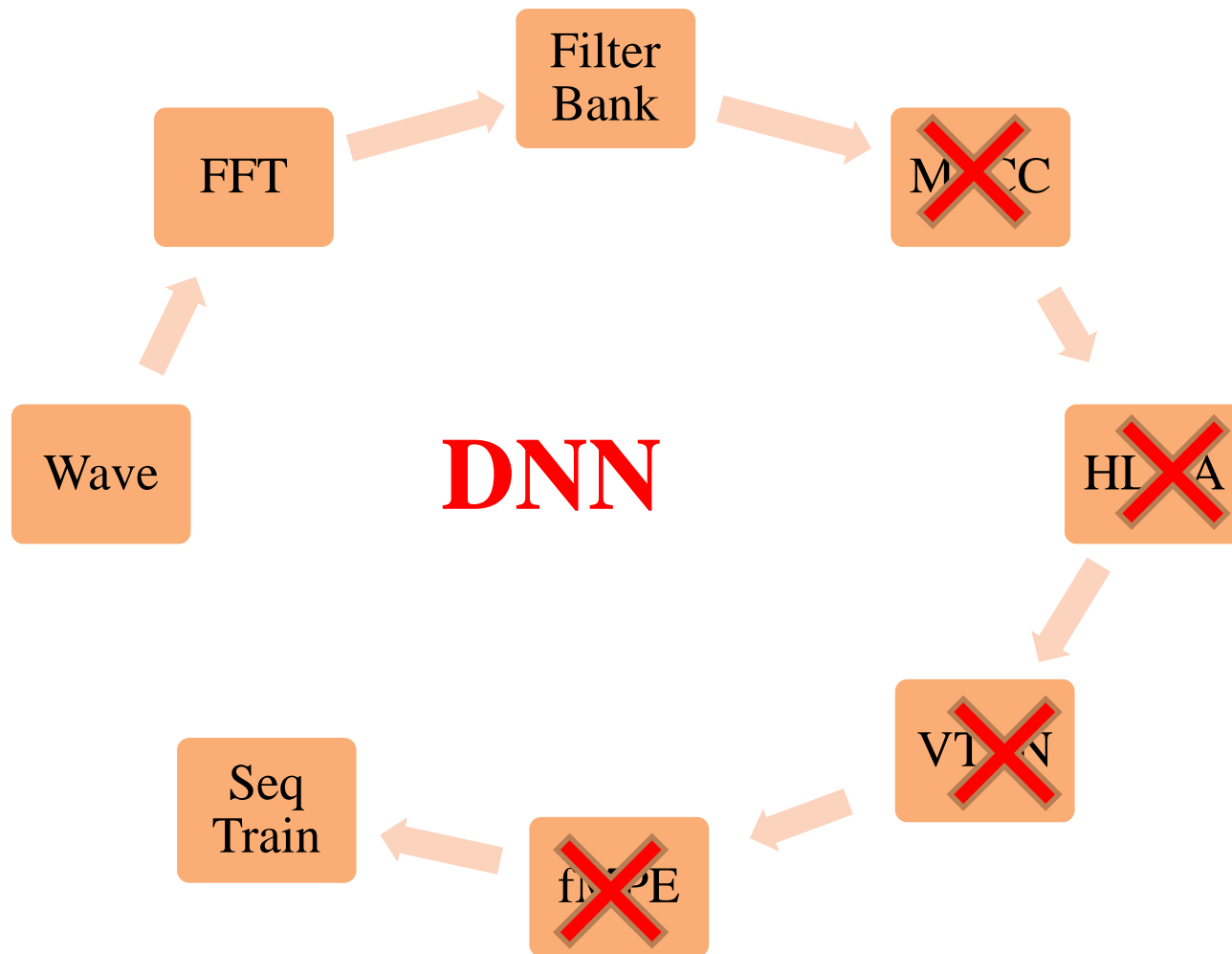
- Bottleneck or Tandem feature extracted from Deep Neural Networks used as features in conventional GMM-HMM
- Convolutional Neural Network
- Deep Tensor Neural Network
- Deep Segmental Neural Network
- Recurrent Neural Network

# To Build a State-Of-the-Art System



CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | **Summary**

# Better Accuracy and Simpler



CD-DNN-HMM | Speed | Invariant Features | Mixed-bandwidth | Multi-lingual | Sequence Train | Adaptation | Summary

# Questions?



# References

---

- X. Chen, A. Eversole, G. Li, D. Yu, and F. Seide (2012), “Pipelined Back-Propagation for Context-Dependent Deep Neural Networks”, Interspeech 2012.
- G. E. Dahl, D. Yu, L. Deng, and A. Acero (2012) "Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition", IEEE Transactions on Audio, Speech, and Language Processing, Jan 2012.
- A-r Mohamed, G. Hinton, G. Penn, (2012) "Understanding how Deep Belief Networks perform acoustic modelling", ICASSP
- G. E. Hinton, S. Osindero, Y. Teh (2006) “A fast learning algorithm for deep belief nets,” Neural Computation, vol. 18, pp. 1527–1554, 2006.
- J.-T. Huang, J. Li, D. Yu, L. Deng, Y. Gong, "Cross-Language Knowledge Transfer Using Multilingual Deep Neural Network With Shared Hidden Layers", ICASSP 2013
- B. Kingsbury, T. N. Sainath, H. Soltau (2012), “Scalable Minimum Bayes Risk Training of Deep Neural Network Acoustic Models Using Distributed Hessian-free Optimization”, Interspeech.
- R. Knies (2012) “Deep-Neural-Network Speech Recognition Debuts”
- R. Knies (2013) “DNN Research Improves Bing Voice Search”
- J. Li, D. Yu, J.-T. Huang, Y. Gong (2012), "Improving Wideband Speech Recognition Using Mixed-Bandwidth Training Data In CD-DDD-HMM", SLT 2012.

# References

---

- G. Li, H. Zhu, G. Cheng, K. Thambiratnam, B. Chitsaz, D. Yu, F. Seide (2012), "Context-Dependent Deep Neural Networks For Audio Indexing Of Real-Life Data", SLT 2012.
- J. Martens (2010), "Deep Learning via Hessian-free Optimization", ICML.
- T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, A.-r. Mohamed (2011) "Making Deep Belief Networks Effective for Large Vocabulary Continuous Speech Recognition", ASRU 2011
- F. Seide, G. Li and D. Yu (2011) "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks", Interspeech 2011, pp. 437-440.
- F. Seide, G. Li, X. Chen, D. Yu (2011) "Feature engineering in context-dependent deep neural networks for conversational speech transcription", ASRU 2011, pp. 24-29.
- A. Senior V. Vanhoucke and M. Z. Mao (2011), "Improving the speed of neural networks on cpus," in Proc. Deep Learning and Unsupervised Feature Learning Workshop, NIPS 2011.
- T. N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, B. Ramabhadran, "Low-Rank Matrix Factorization For Deep Neural Network Training With High-Dimensional Output Targets", ICASSP 2014
- T. Simonite (2012), "Google Puts Its Virtual Brain Technology to Work".

# References

---

- M. Seltzer, D. Yu, Y. Wang, "An Investigation Of Deep Neural Networks For Noise Robust Speech Recognition", ICASSP 2013
- H. Su, G. Li, D. Yu, F. Seide, "Error Back Propagation For Sequence Training Of Context-Dependent Deep Networks For Conversational Speech Transcription", ICASSP 2013
- D. Yu, L. Deng, F. Seide (2012), "Large Vocabulary Speech Recognition Using Deep Tensor Neural Networks", Interspeech 2012.
- D. Yu, F. Seide, G. Li, L. Deng (2012), "Exploiting Sparseness In Deep Neural Networks For Large Vocabulary Speech Recognition", ICASSP 2012
- D. Yu, S. Siniscalchi, L. Deng, C.-H. Lee (2012), "Boosting Attribute And Phone Estimation Accuracies With Deep Neural Networks For Detection-Based Speech Recognition", ICASSP 2012, pp. 4169-4172.
- D. Yu, K. Yao, H. Su, G. Li, F. Seide, "KL-Divergence Regularized Deep Neural Network Adaptation For Improved Large Vocabulary Speech Recognition", ICASSP 2013
- D. Yu, M. L. Seltzer, J. Li, J.-T. Huang, F. Seide, "Feature Learning in Deep Neural Networks - Studies on Speech Recognition Tasks", ICLR 2013.