YORK UNIVERSITÉ UNIVERSITY redefine THE POSSIBLE.

Why DNN Works for Acoustic Modeling in Speech Recognition?

Prof. Hui Jiang

Department of Computer Science and Engineering

York University, Toronto, Ont. M3J 1P3, CANADA

Joint work with Y. Bao, J. Pan, O. Abdel-Hamid

Outline

YORK UNIVERSITÉ UNIVERSITY

redefine THE POSSIBLE.

- Introduction
- DNN/HMM for Speech
- Bottleneck Features
- Incoherent Training
- Conclusions
- Other DNN Projects

This talk is based on the following two papers:

[1] J. Pan, C. Liu, Z. Wang, Y. Hu and H. Jiang, ``Investigations of Deep Neural Networks for Large Vocabulary Continuous Speech Recognition", *Proc. of International Symposium on Chinese Spoken Language Processing (ISCSLP'2012)*, Hong Kong, December 2012.

[2] Y. Bao, H. Jiang, L. Dai, C. Liu, "Incoherent Training of Deep Neural Networks to De-correlated Bottleneck Features for Speech Recognition," submitted to 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'13), Vancouver, Canada.

Introduction: ASR History

- ASR formulation:
  - GMM/HMM + n-gram + Viterbi search
- Technical advances (incremental) over past 10 years:
  - Adaptation (speaker/environment): 5% *rel.* gain
  - Discriminative Training: 5-10% *rel.* gain
  - Feature normalization: 5% *rel.* gain
  - ROVER: 5% *rel.* gain
- More and more data ➔ better and better accuracy
  - read speech (>90%), telephony speech (>70%)
  - meeting/voicemail recording (<60%)

Acoustic Modeling: Optimization

- Acoustic modeling → large-scale optimization
  - 2000+ hour data → GMMs/HMM

    billions of samples → 100+ million free parameters

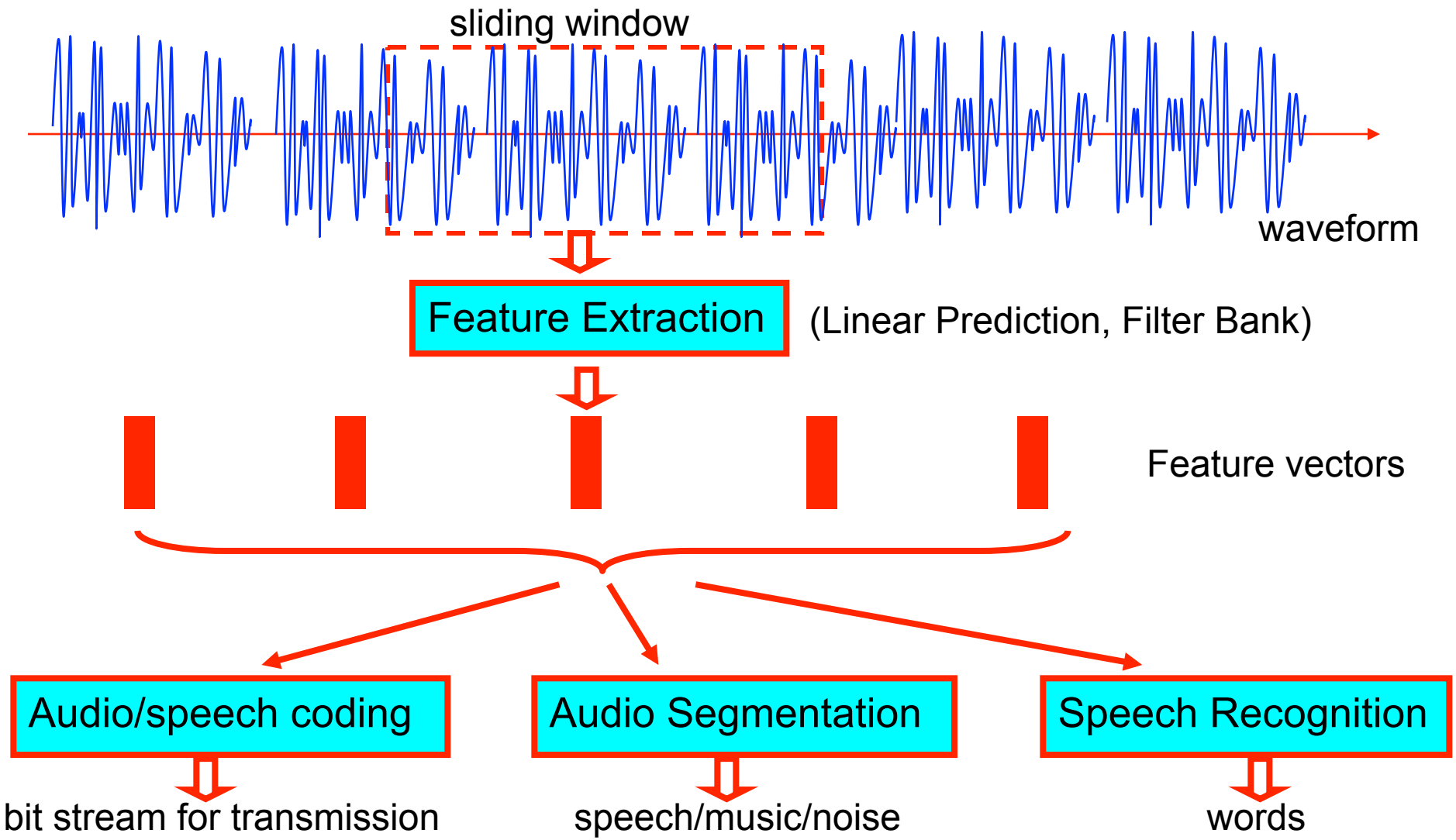
- Training Methods
  - Maximum Likelihood Estimation (MLE)
  - Discriminative Training (DT)

- Engineering Issues
  - Efficiency: feasible with 100-1000 of CPUs
  - Reliability: robust estimation of all parameters

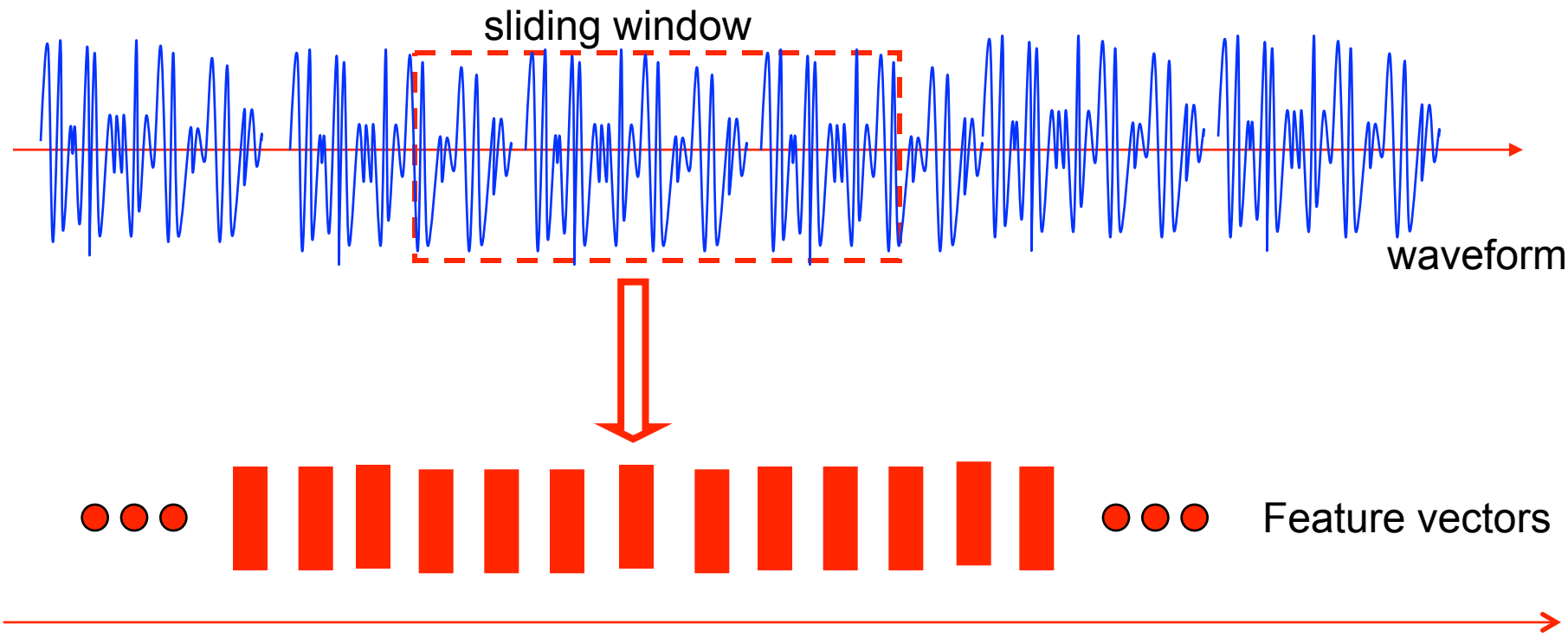
Neural Network for ASR

- **1990s: MLP for ASR** *(Bourlard and Morgan, 1994)*
  - **NN/HMM hybrid model (worse than GMM/HMM)**
- **2000s: TANDEM** *(Hermansky, Ellis, et al., 2000)*
  - **Use MLP as Feature Extraction (5-10% rel. gain)**
- **2006: DNN for small tasks** *(Hinton et al., 2006)*
  - **RBM-based pre-training for DNN**
- **2010: DNN for small-scale ASR** *(Mohamed, Yi, et al. 2010)*
- **2011: DNN for large-scale ASR**
  - **Over 30% rel. gain in Switchboard** *(Seide et al., 2011)*

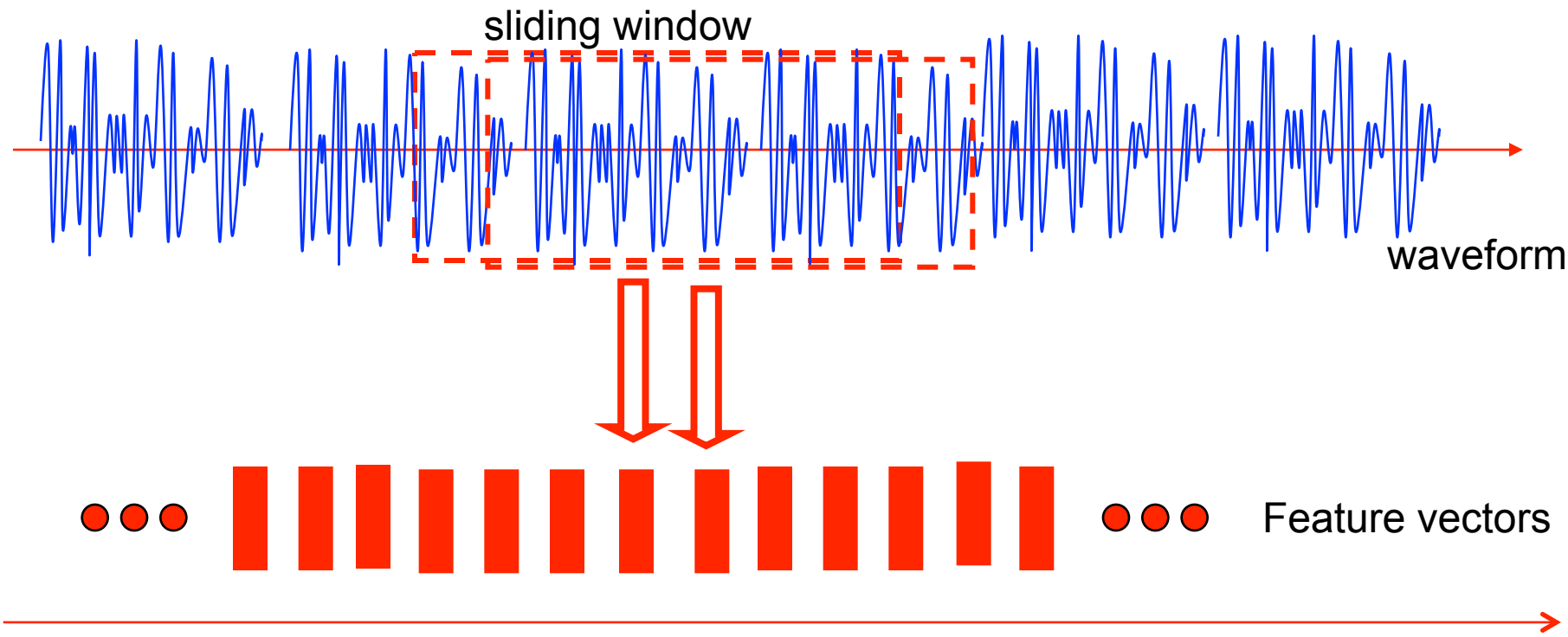
ASR Frontend



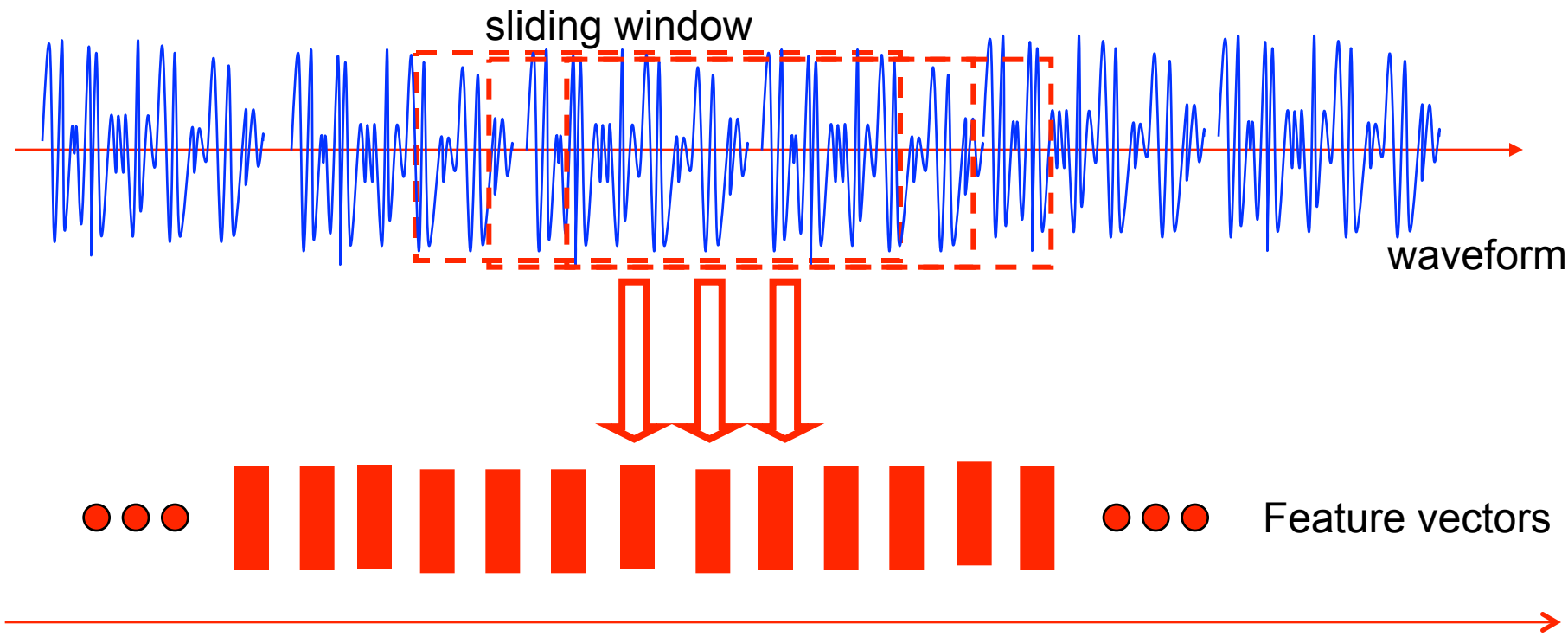
Short-time Analysis



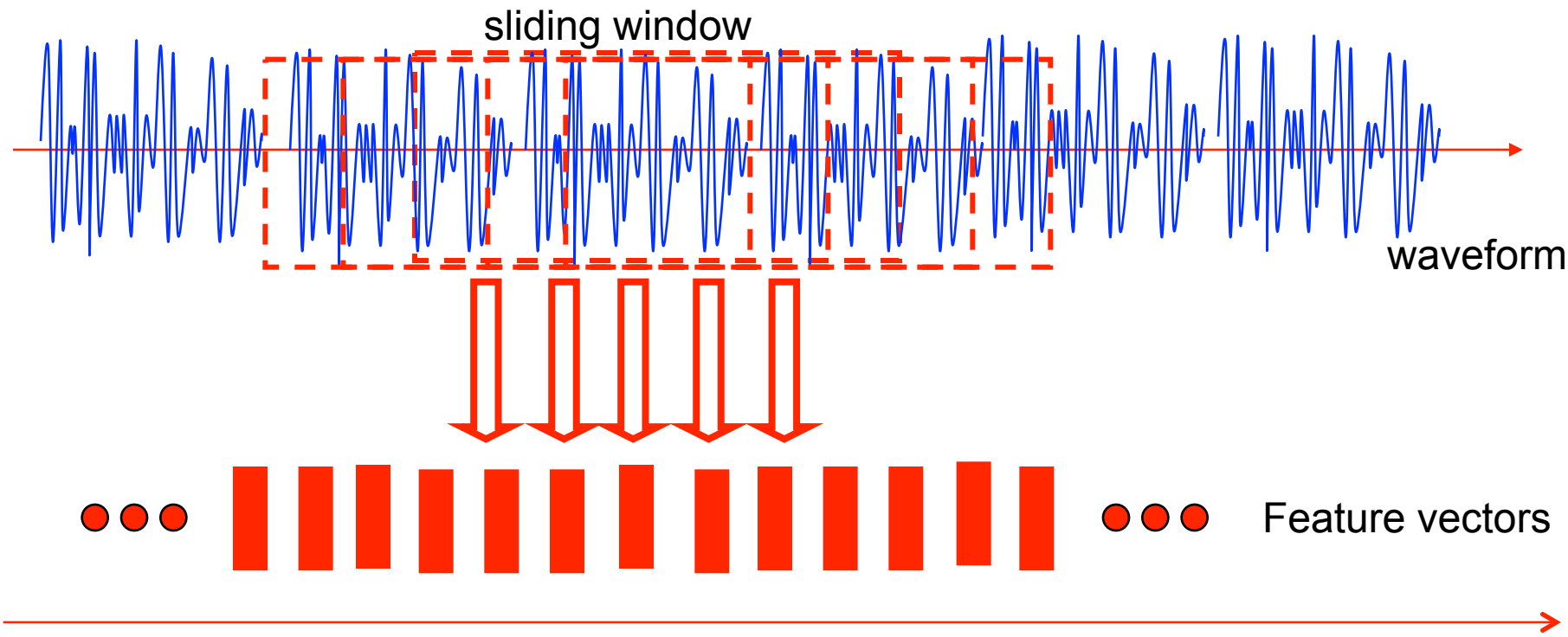
Short-time Analysis



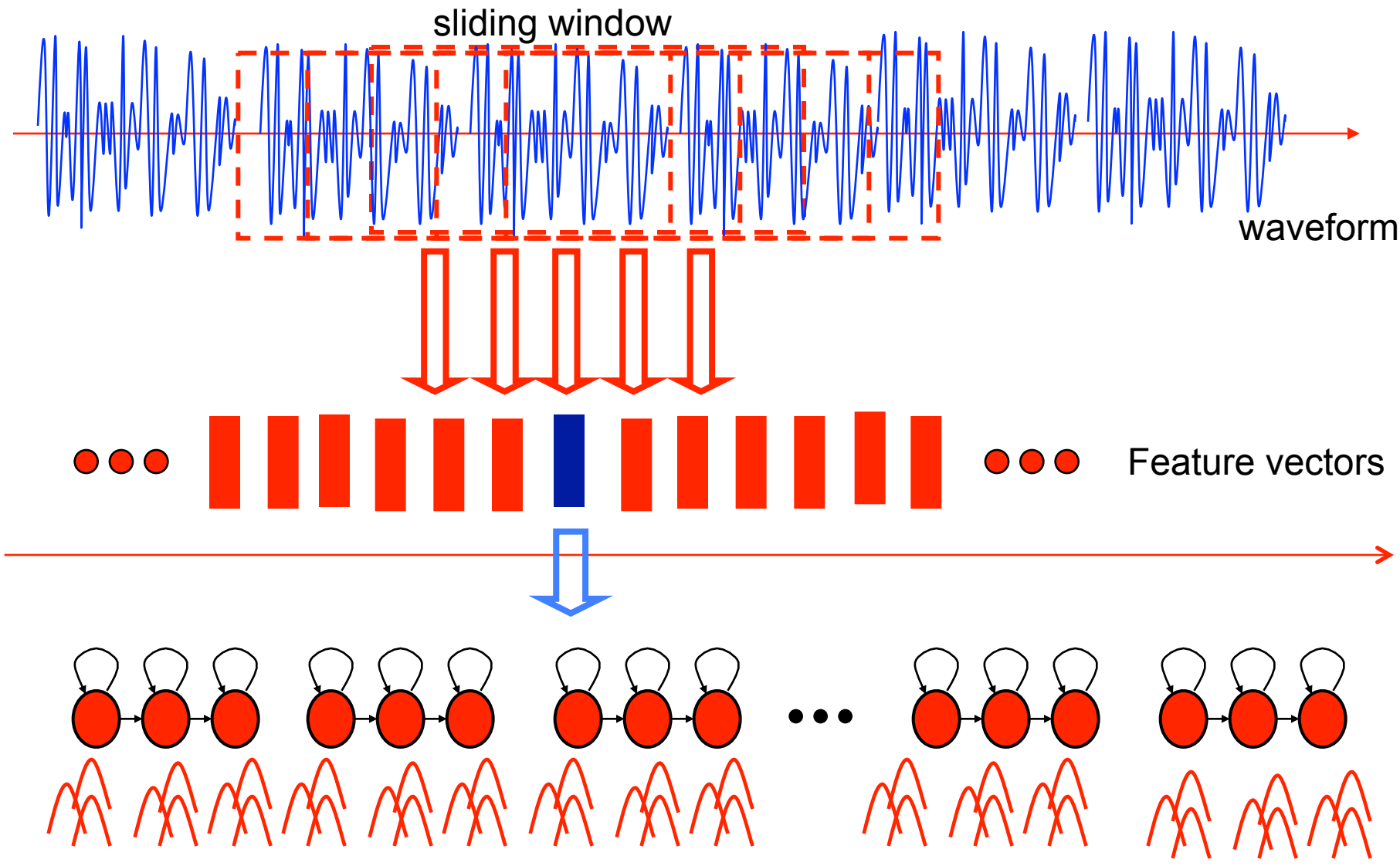
Short-time Analysis



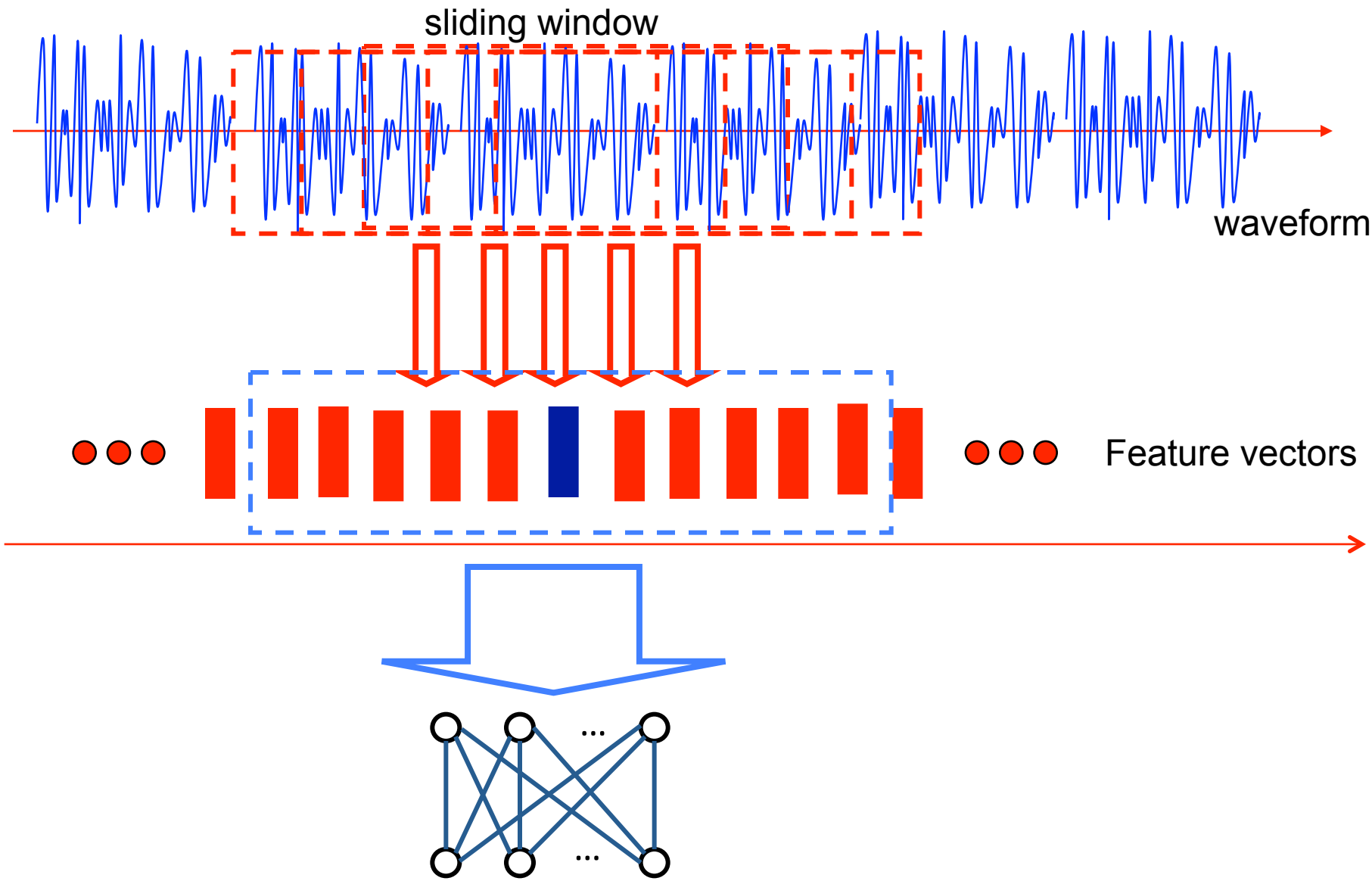
Short-time Analysis



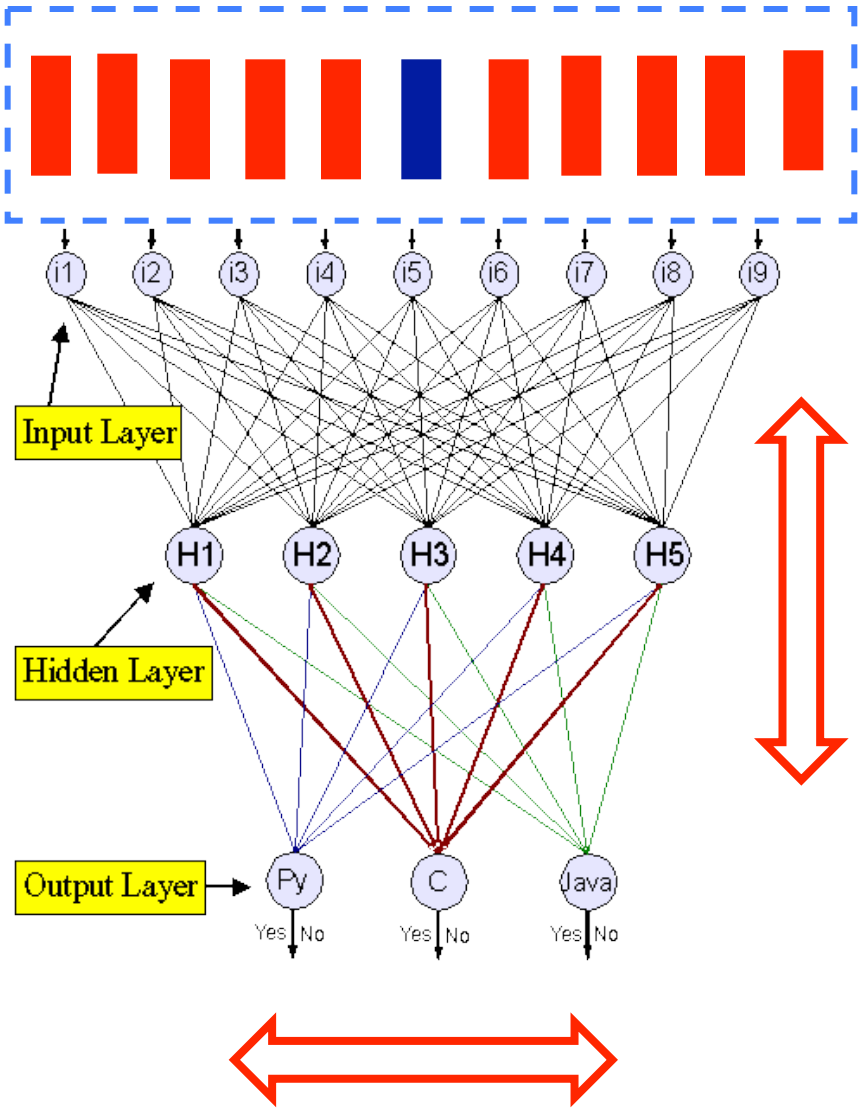
ASR Frontend: GMM/HMM



ASR Frontend: NN/HMM



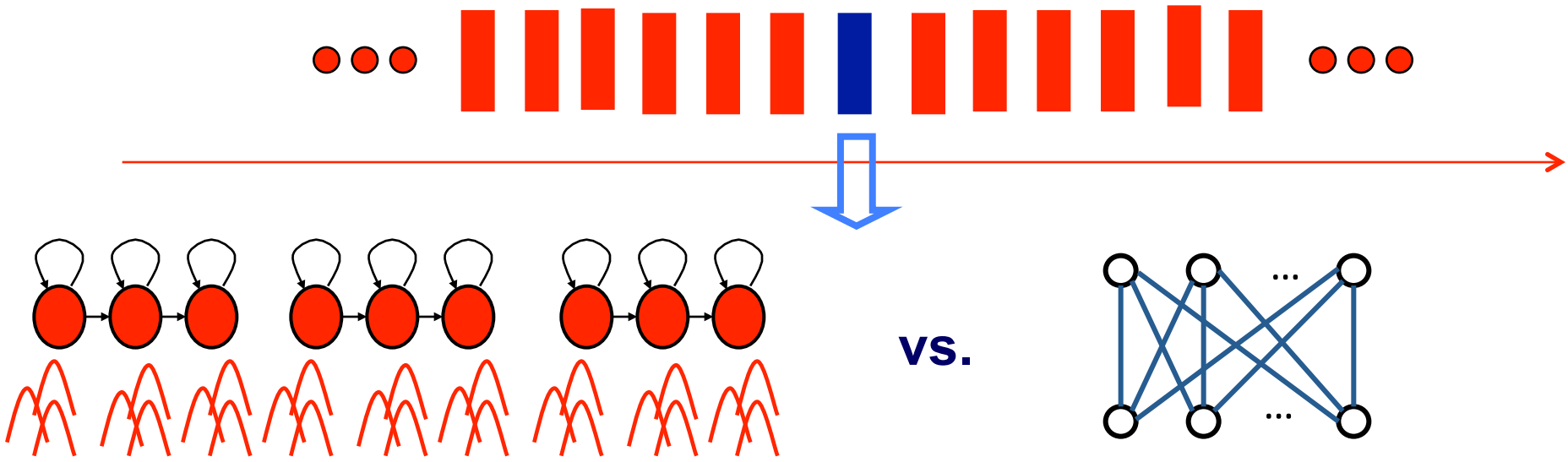
NN for ASR: old and new



- **Deeper network**
  - **more hidden layers**
  - **(1 → 6-7 layers)**
- **Wider network**
  - **More hidden nodes**
  - **More output nodes**
  - **(100 → 5-10 K)**
- **More data**
  - **10-20 hours → 2-10 k hours training data**

GMMs/HMM vs. DNN/HMM

- Different acoustic models
  - GMMs vs. DNN
- Different feature vectors
  - 1 frame vs. concatenated frames (11-15 frames)



Experiment (I): GMMs/HMM vs. DNN/HMM

- 70-hour Chinese ASR task; 4000 tied HMM states
- GMM: 30 Gaussians per state
- DNN: pre-trained; 1024 nodes per layer; 1-6 hidden layers

Numbers in word error rates (%)
NN-1: 1 hidden layer; DNN-6: 6 hidden layers
MPE GMM: discriminatively trained GMM/HMM

Context window	NN-1	DNN-6	MPE GMM
1	*18.0*	*17.0*	*16.7*
Context window	3	5	7
DNN-6	*14.2*	*13.7*	*13.5*
Context window	9	11	13
DNN-6	*13.5*	***13.4***	*13.6*

Experiment (II): GMMs/HMM vs. DNN/HMM

- 320-hour English Switchboard task; 9000 tied HMM states
- GMM: 40 Gaussian per state
- DNN: pre-trained; 2000 nodes per layer; 1-5 hidden layers

Numbers in word error rates (%)
NN-1: 1 hidden layer; DNN-3/5: 3/5 hidden layers
MPE GMM: discriminatively trained GMM/HMM

Context window		NN-1	DNN-3	DNN-5	MPE GMM
1	Hub98	44.8	43.9	42.6	43.4
	Hub01	35.4	34.8	33.1	32.8
11	Hub98	39.7	33.4	**31.2**	n/a
	Hub01	31.4	25.6	**23.7**	n/a

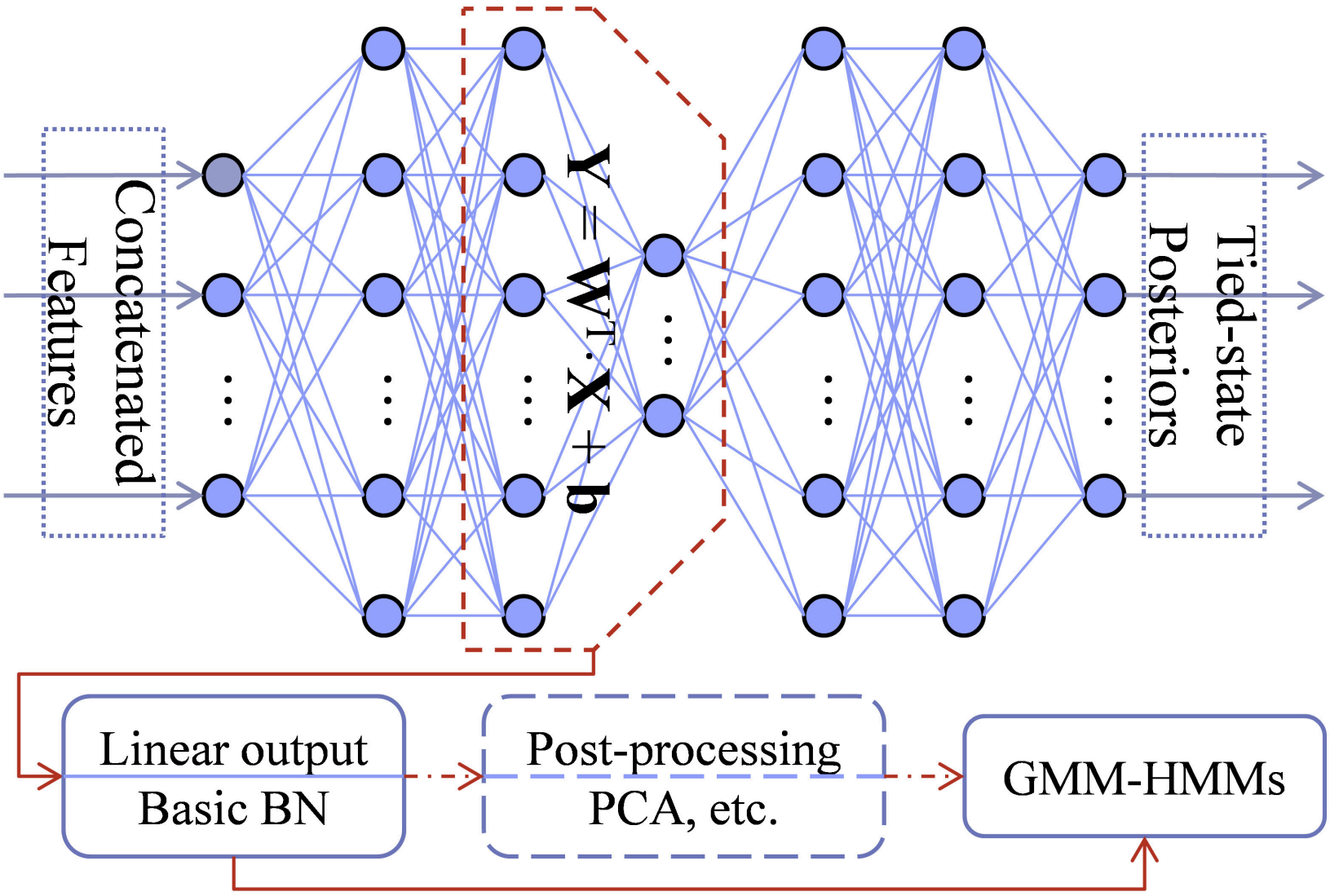
Conclusions (I)

- The gain of DNN/HMM hybrid is almost entirely attributed to the concatenated frames.
  - The concatenated features contain almost all additional information resulting in the gain.
  - But they are highly correlated.
- DNN is powerful to leverage highly correlated features.

What's next

- How about GMM/HMM?
- Hard to explore highly correlated features in GMMs.
  - Requires dimensional reduction for de-correlation.
- Linear dimensional reduction (PCA, LDA, KLT, …)
  - Failed to compete with DNN.
- Nonlinear dimensional reduction
  - Using NN/DNN *(Hinton et al.)*, a.k.a. bottleneck features
  - Manifold learning, LLE, MDS, SNE, …?

Bottleneck (BN) Feature



Experiment (I): Bottleneck(BN) Features

70-hour Chinese ASR Task (word error rate in %)
MLE: maximum likelihood estimation
MPE: discriminative training

	MLE	MPE
GMM-HMM	18.2	**16.7**
CD-DNN-HMM	**13.1**	
BN-GMM-HMM	13.6	**12.2**

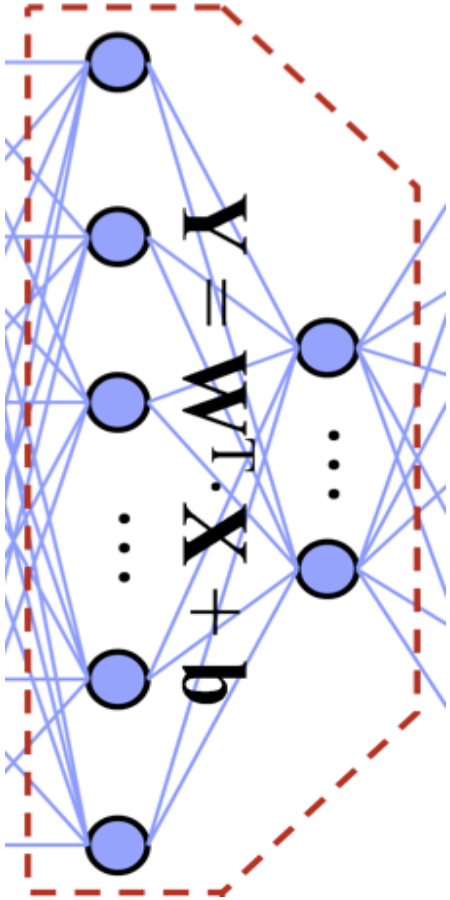
Experiment (II): Bottleneck Features (BN)

320-hour English Switchboard Task (word error rate in %)
MLE: maximum likelihood estimation
MPE: discriminative training

Acoustic Models	Hub5e98		Hub5e01	
	MLE	MPE	MLE	MPE
GMM-HMMs	46.6	**43.4**	35.4	**32.8**
CD-DNN-HMMs	**31.2**		**23.7**	
BN-GMM-HMMs	34.3	**31.3**	26.0	**23.2**

Incoherent Training

- Bottleneck (BN) works but:
  - BN hurts DNN performance a little
  - Increasing BN ➔ correlation up
- Can we do better?
- The Idea: embedding de-correlation into back-propagation of DNN training.
  - De-correlation by constraining columns of weight matrix *W*
  - How to constrain?



Incoherent Training

- **Define coherence of DNN weight matrix *W* as:**

$$G_W = \max_{i,j} g_{ij} = \max_{i,j} \frac{\left| w_i \cdot w_j \right|}{\left\| w_i \right\| \left\| w_j \right\|}$$

- **A matrix with smaller coherence indicates all of its column vectors are less similar.**

- **Approximate coherence using soft-max:**

$$G_W = \log \left(\frac{1}{M} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \exp\{\beta \cdot g_{ij}\} \right)^{\frac{1}{\beta}}$$

Incoherent Training

- **All DNN weight matrices are optimized by minimizing a regularized objective function:**

$$F^{(new)} = F^{(old)} + \alpha \cdot \max_{W} G_W$$

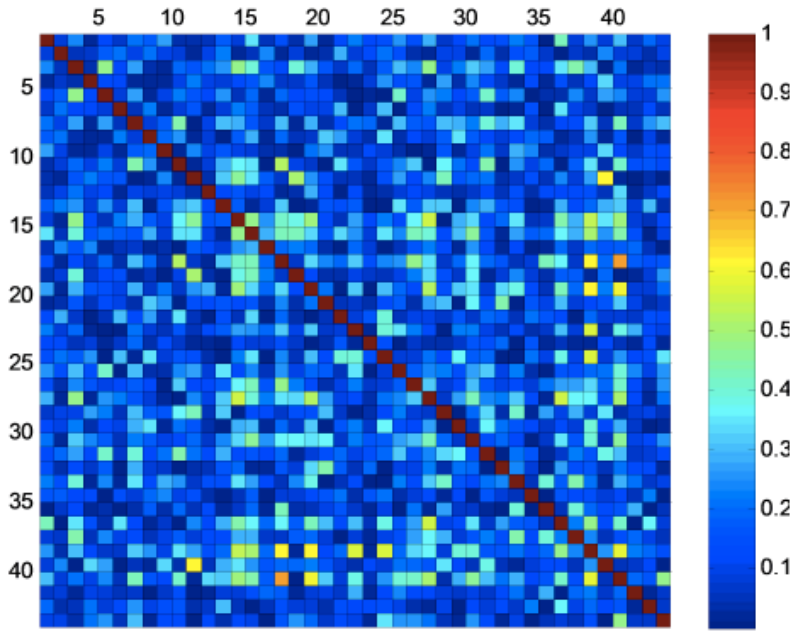
- **Derivatives of coherence:**

$$\frac{\partial G_W}{\partial w_k} = \sum_{j=1}^{N} \gamma_{kj} g_{kj} \left[\frac{\mathbf{w}_j}{\mathbf{w}_k \cdot \mathbf{w}_j} - \frac{\mathbf{w}_k}{\mathbf{w}_k \cdot \mathbf{w}_k} \right]$$

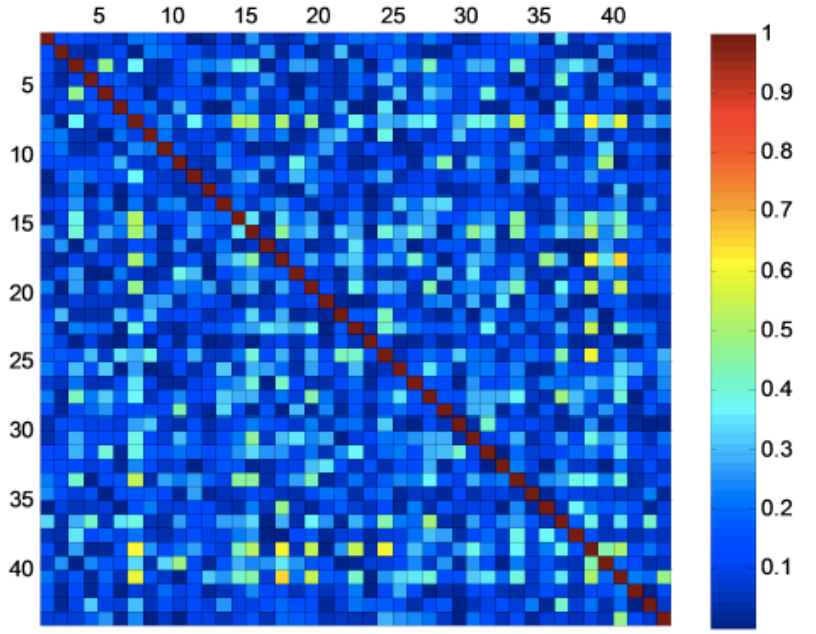
- **Back-propagation is still applicable…**

Incoherent Training: De-correlation

Applying incoherent training to one weight matrix in BN



(a) Baseline BN



(b) Weight-matrix Incoherent BN

Incoherent Training: Data-driven

- If only applying to one weight matrix *W*: $Y = W^T X + b$
- Covariance matrix of Y: $C_Y = W^T C_X W$
- Directly measure correlation coefficients based on the above covariance matrix:

$$G_W = \max_{i,j} g_{ij}$$

with

$$g_{ij} = \frac{\left|(\mathbf{w}_i)^\top \mathbf{C_X} \mathbf{w}_j\right|}{\sqrt{(\mathbf{w}_i)^\top \mathbf{C_X} \mathbf{w}_i} \cdot \sqrt{(\mathbf{w}_j)^\top \mathbf{C_X} \mathbf{w}_j}}$$

Cx is estimate from one mini-batch each time

Incoherent Training: Data-driven

- **After *soft-max*, Derivatives can be computed as:**

$$\frac{\partial G_W}{\partial w_k} = \frac{\sum_{j \neq k}^{N} \left[\exp\{\beta \cdot g_{kj}\} \cdot \frac{\partial g_{kj}}{\partial \mathbf{w}_k} \right]}{\sum_{i=1}^{N} \sum_{j=i+1}^{N} \exp\{\beta \cdot g_{ij}\}}$$

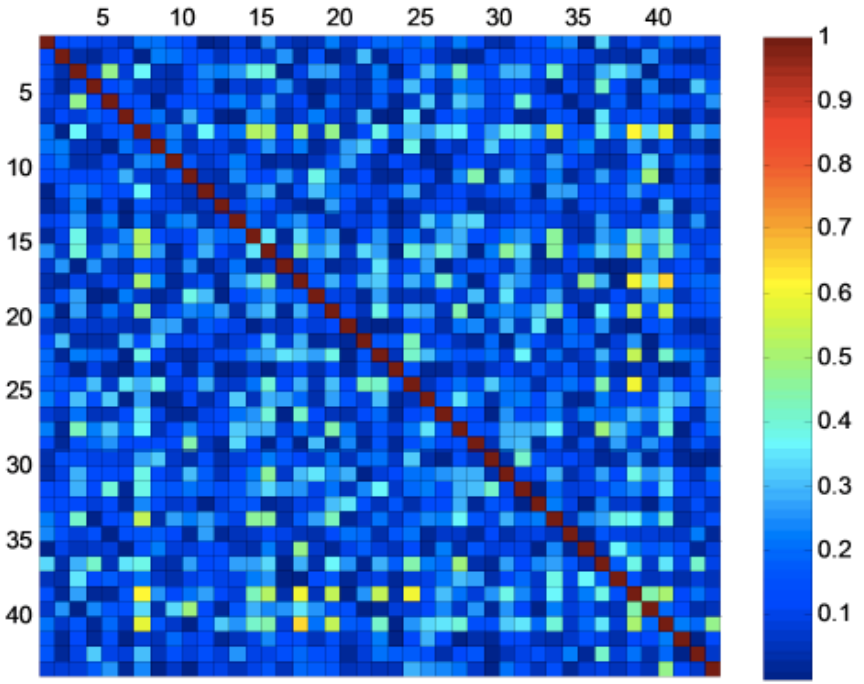
where

$$\frac{\partial g_{kj}}{\partial \mathbf{w}_k} = g_{kj} \left[\frac{\mathbf{C_X w}_j}{(\mathbf{w}_k)^\top \mathbf{C_X w}_j} - \frac{\mathbf{C_X w}_k}{(\mathbf{w}_k)^\top \mathbf{C_X w}_k} \right]$$

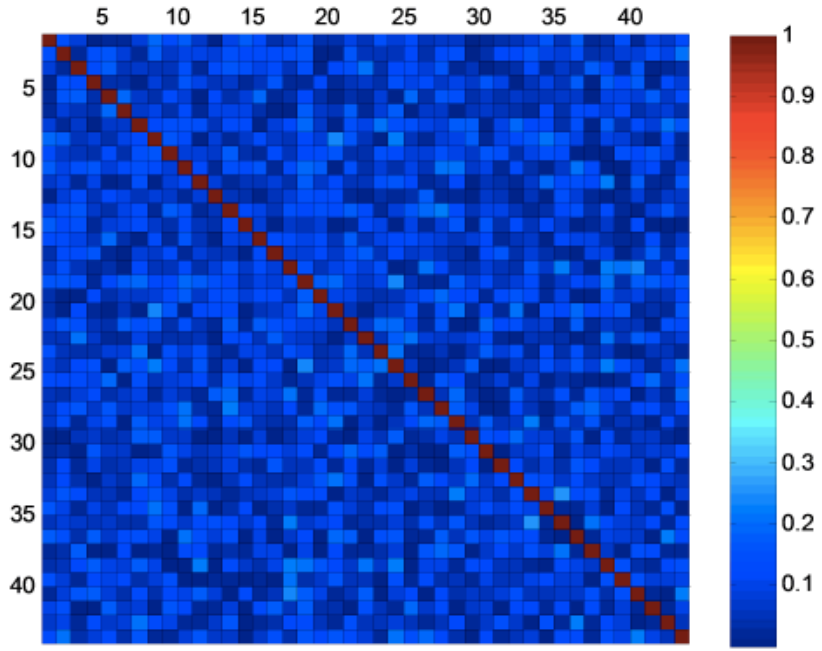
- **Back-propagation still applies except Cx is computed for each mini-batch**

Incoherent Training: Data-driven De-correlation

When applying Incoherent Training to one weight matrix



(b) Weight-matrix Incoherent BN



(c) Mini-batch-data Incoherent BN

Experiment (I): Incoherent Training

70-hour Chinese ASR Task (word error rate in %)
MLE: maximum likelihood estimation
MPE: discriminative training
DNN-HMM: 13.1%

	MLE	MPE
Baseline BN	13.6	**12.2**
Weight-matrix Incoherent BN	**13.3**	-
Mini-batch-data Incoherent BN	**13.2**	**12.0**

Experiment (II): Incoherent Training

320-hour English Switchboard Task (word error rate in %)
MLE: maximum likelihood estimation
MPE: discriminative training
DNN-HMM: 31.2% (Hub5e98) and 23.7% (Hub5e01)

BN Feature Systems	Hub5e98		Hub5e01	
	MLE	MPE	MLE	MPE
Baseline BN	34.3	31.3	26.0	23.2
Weight-matrix Incoherent BN	**34.0**	-	**25.7**	-
Mini-batch-data Incoherent BN	**33.8**	**31.0**	**25.6**	**22.8**

Conclusions (II)

- Possible to compete with DNN under the traditional GMMs/HMM framework.
- Promising to use bottleneck features learned from the proposed incoherent training.
- Benefits over DNN/HMM:
  - Slightly better performance
  - Enjoy other ASR techniques (adaptation, …)
  - Faster training process
  - Faster decoding process

Future works

- Apply incoherent training to general DNN learning
- Other nonlinear dimensional reduction methods for concatenated features