Deep Neural Networks for Acoustic Modeling in Speech Recognition

Presented by Peidong Wang

04/04/2016

Hinton, Geoffrey, et al. “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups.” *Signal Processing Magazine, IEEE* 29.6 (2012): 82-97.

Content

- Speech Recognition System
- GMM-HMM Model
- Training Deep Neural Networks
- Generative Pretraining
- Experiments
- Discussion

Content

- Speech Recognition System
- GMM-HMM Model
- Training Deep Neural Networks
- Generative Pretraining
- Experiments
- Discussion

Speech Recognition System

- **Goal**
- Converting speech to text

- **A Mathematical Perspective**

$$\hat{w} = \arg\max_{w} \{P(w \mid Y)\}$$

or

$$\hat{w} = \arg\max_{w} \{P(Y \mid w)P(w)\}$$

Content

- Speech Recognition System
- GMM-HMM Model
- Training Deep Neural Networks
- Generative Pretraining
- Experiments
- Discussion

GMM-HMM Model

- **GMM and HMM**
- GMM is short for Gaussian Mixture Model, and HMM is short for Hidden Markov Model.

- **Predecessor of DNNs**
- Before Deep Neural Networks (DNNs), the most commonly used speech recognition systems were consisted of GMMs and HMMs.

GMM-HMM Model

- **HMM**
- HMM is used to deal with the temporal variability of speech.

- **GMM**
- GMM is used to represent the relationship between HMM states and the acoustic input.

GMM-HMM Model

- **Features**
- The features is typically represented by concatenating Mel-frequency cepstral coefficients (MFCCs) or perceptual linear predictive coefficients (PLPs) computed from the raw waveform and their first- and second-order temporal differences.

GMM-HMM Model

- **Shortcoming**
- GMM-HMM models are statistically inefficient for modeling data that lie on or near a nonlinear manifold in the data space.
- For example, modeling the set of points that lie very close to the surface of a sphere.

Content

- Speech Recognition System
- GMM-HMM Model
- Training Deep Neural Networks
- Generative Pretraining
- Experiments
- Discussion

Training Deep Neural Networks

- **Deep Neural Network (DNN)**
- A DNN is a feed-forward, artificial neural network that has more than one layer of hidden units between its inputs and its outputs.
- With nonlinear activation functions, DNN is able to model an arbitrary nonlinear function (projection from inputs to outputs). [*]

[*] Added by the presenter.

Training Deep Neural Networks

- **Activation Function of the Output Units**
- The activation function of the output units is “softmax” function.
- The mathematical expression is as follows.

$$p_j = \frac{\exp(x_j)}{\sum_k \exp(x_k)}$$

Training Deep Neural Networks

- **Objective Function**
- When using the softmax output function, the natural objective function (cost function) C is the cross-entropy between the target probabilities d and the outputs of the softmax, p.
- The mathematical expression is as follows.

$$C = \sum_j d_j \log p_j$$

Training Deep Neural Networks

- **Weight Penalties and Early Stopping**
- To reduce overfitting, large weights can be penalized in proportion to their squared magnitude, or the learning can simply be terminated at the point which performance on a held-out validation set starts getting worse.

Training Deep Neural Networks

- **Overfitting Reduction**
- Generally speaking, there are three methods.
- *Weight penalties and early stopping* can reduce the overfitting but only by removing much of the modeling power.
- *Very large training sets* can reduce overfitting but only by making training very computationally expensive.
- *Generative Pretraining*

Content

- Speech Recognition System
- GMM-HMM Model
- Training Deep Neural Networks
- Generative Pretraining
- Experiments
- Discussion

Generative Pretraining

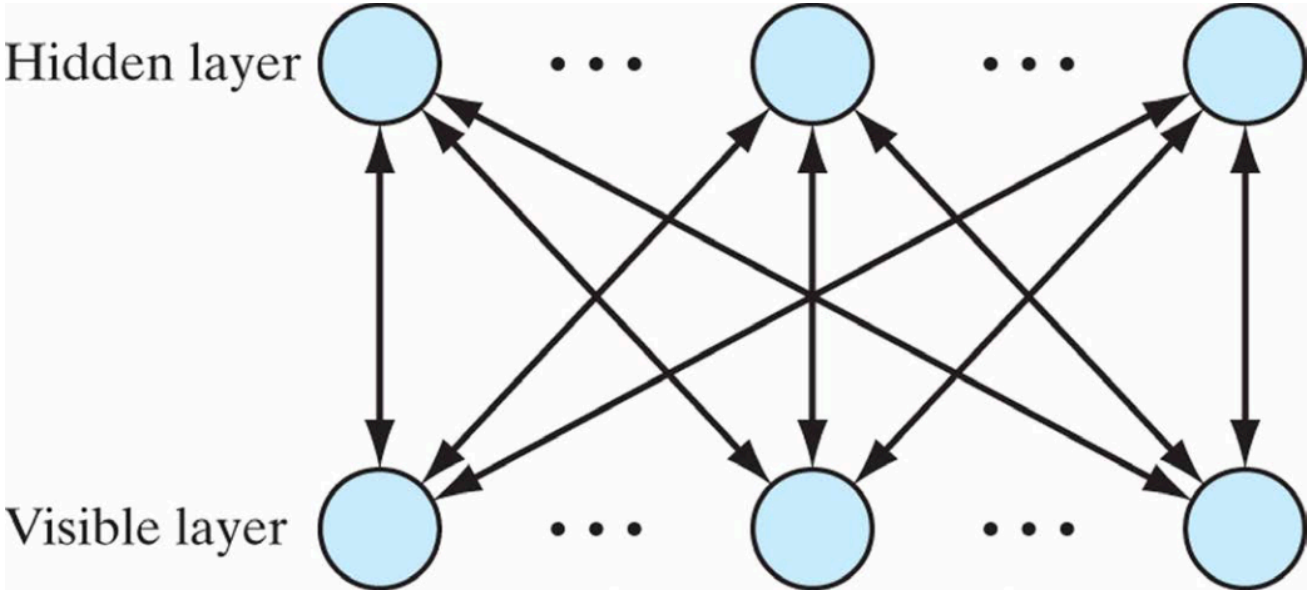
- **Purpose**
- The multiple layers of feature detectors (the result of this step) can be used as a good starting point for a discriminative “fine-tuning” phase during which backpropagation through the DNN slightly adjusts the weights and improves the performance.
- In addition, this step can significantly reduce overfitting.

Generative Pretraining

- **Restricted Boltzmann Machine (RBM)**
- RBM consists of a layer of stochastic binary “visible” units that represent binary input data connected to a layer of stochastic binary hidden (latent) units that learn to model significant nonindependencies between the visible units.
- There are undirected connections between visible and hidden units but no visible-visible or hidden-hidden connections.

Generative Pretraining

- **Restricted Boltzmann Machine (RBM) (Cont'd)**
- The framework of an RBM is shown below.



From: Slides in CSE5526 Neural Networks

Generative Pretraining

- **Restricted Boltzmann Machine (RBM) (Cont'd)**
- RBM uses a single set of parameters, **W**, to define the joint probability of a vector of values of the observable variables, **v**, and a vector of values of the latent variables, **h**, via an energy function, E.

$$p(v,h;W)=\frac{1}{Z}e^{-E(v,h;W)},Z=\sum_{v',h'}e^{-E(v',h';W)}$$

$$E(v,h)=-\sum_{i\in visible}a_i v_i-\sum_{j\in visible}b_j h_j-\sum_{i,j}v_i h_j w_{ij}$$

Generative Pretraining

- **Restricted Boltzmann Machine (RBM) (Cont'd)**
- The probability that the network assigns to a visible vector, v, is given by summing over all possible hidden vectors.

$$p(v) = \frac{1}{Z} \sum_{h} e^{-E(v,h)}$$

- The derivative of the log probability of a training set with respect to a weight is surprisingly simple. The angle brackets denote expectations under the corresponding distribution.

$$\frac{1}{N} \sum_{n=1}^{N} \frac{\partial \log p(v^n)}{\partial w_{ij}} = < v_i h_j >_{data} - < v_i h_j >_{model}$$

Generative Pretraining

- **Restricted Boltzmann Machine (RBM) (Cont'd)**
- The learning rule is thus as follows.

$$\Delta w_{ij} = \varepsilon(< v_i h_j >_{data} - < v_i h_j >_{model})$$

- A better learning procedure is contrastive divergence (CD), which is shown below. The subscript "recon" denotes a step in CD when the states of visible units are assigned 0 or 1 according to the current states of the hidden units.

$$\Delta w_{ij} = \varepsilon(< v_i h_j >_{data} - < v_i h_j >_{recon})$$

Generative Pretraining

- **Modeling Real-Valued Data**
- Real-valued data, such as MFCCs, are more naturally modeled by linear variables with Gaussian noise and the RBM energy function can be modified to accommodate such variables, giving a Gaussian-Bernoulli RBM (GRBM).

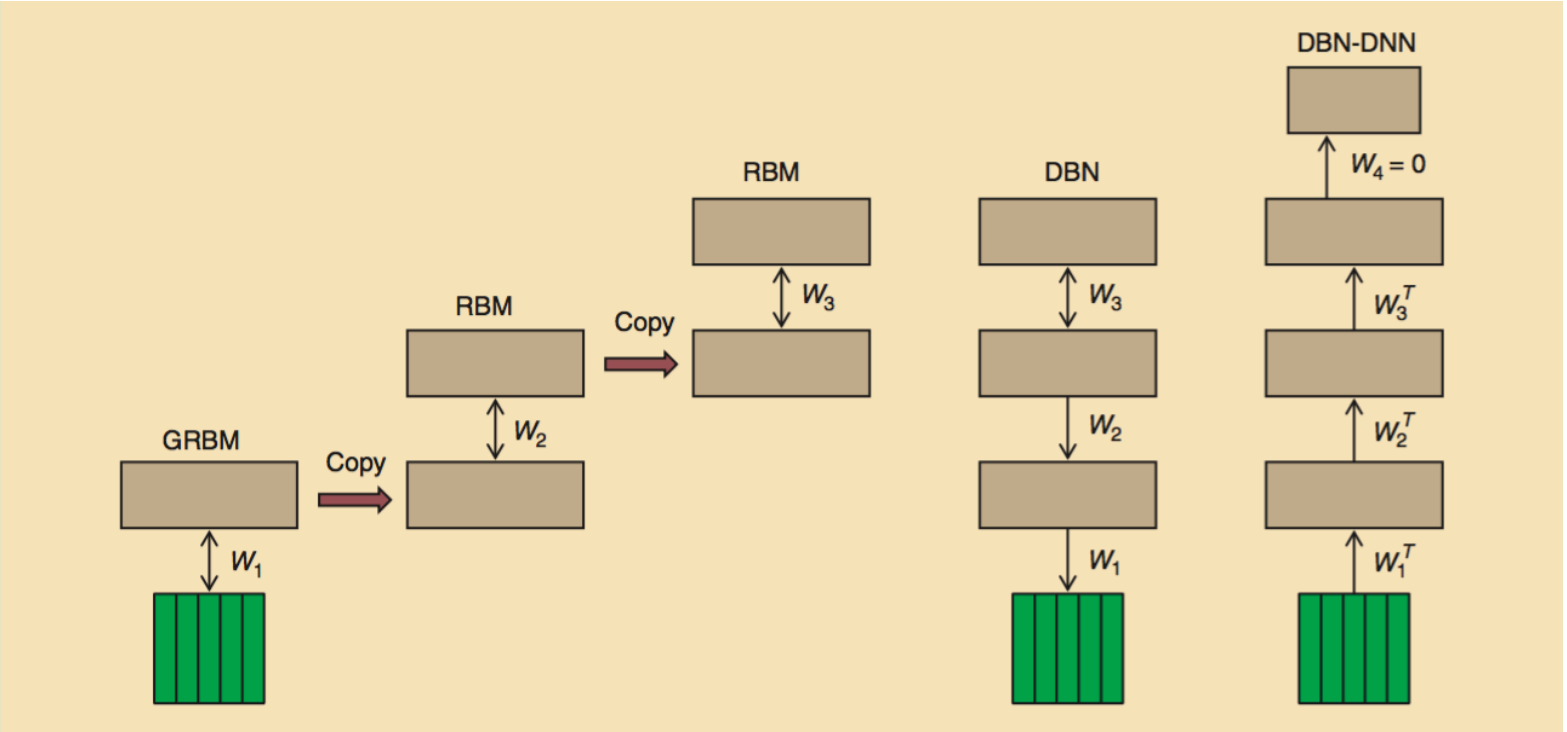
$$E(v,h) = \sum_{i \in vis} \frac{(v_i - a_i)^2}{2\sigma_i^2} - \sum_{j \in hid} b_j h_j - \sum_{i,j} \frac{v_i}{\sigma_i} h_j w_{ij}$$

Generative Pretraining

- **Stacking RBMs to Make a Deep Belief Network**
- After training an RBM on the data, the inferred states of the hidden units can be used as data for training another RBM that learns to model the significant dependencies between the hidden units of the first RBM.
- This can be repeated as many times as desired to produce many layers of nonlinear feature detectors that represent progressively more complex statistical structure in the data.

Generative Pretraining

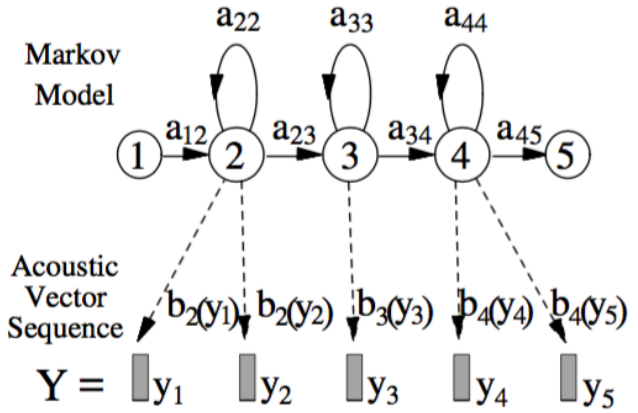
- **Stacking RBMs to Make a Deep Belief Network (Cont'd)**



From: The paper

Generative Pretraining

- **Interfacing a DNN with an HMM**
- In an HMM framework, the hidden variables denote the states of the phone sequence, and the "visible" variables denote the feature vectors. [*]



[*] Added by the presenter

From: Gales, Mark, and Steve Young. "The application of hidden Markov models in speech recognition." *Foundations and trends in signal processing* 1.3 (2008): 195-304.

Generative Pretraining

- **Interfacing a DNN with an HMM (Cont'd)**
- To compute a Viterbi alignment or to run the forward-backward algorithm within the HMM framework, we require the likelihood p(AcousticInput|HMMstate).
- A DNN, however, outputs probabilities of the form p(HMMstate|AcousticInput).

Generative Pretraining

- **Interfacing a DNN with an HMM (Cont'd)**
- The posterior probabilities that the DNN outputs can be converted into the scaled likelihood by dividing them by the frequencies of the HMM states in the forced alignment that is used for fine-tuning the DNN.
- *Forced alignment* is a procedure used to generate labels for the training process. [*]

[*] Added by the presenter

Generative Pretraining

- **Interfacing a DNN with an HMM (Cont'd)**
- All of the likelihoods produced in this way are scaled by the same unknown factor of p(AcousticInput).
- Although this appears to have little effect on some recognition tasks, it can be important for tasks where training labels are highly unbalanced.

Content

- Speech Recognition System
- GMM-HMM Model
- Training Deep Neural Networks
- Generative Pretraining
- Experiments
- Discussion

Experiments

- **Phonetic Classification and Recognition on TIMIT**
- The TIMIT data set is a relatively small data set which provides a simple and convenient way of testing new approaches to speech recognition.

Experiments

- **Phonetic Classification and Recognition on TIMIT (Cont'd)**

[TABLE 1] COMPARISONS AMONG THE REPORTED SPEAKER-INDEPENDENT (SI) PHONETIC RECOGNITION ACCURACY RESULTS ON TIMIT CORE TEST SET WITH 192 SENTENCES.

METHOD	PER
CD-HMM [26]	27.3%
AUGMENTED CONDITIONAL RANDOM FIELDS [26]	26.6%
RANDOMLY INITIALIZED RECURRENT NEURAL NETS [27]	26.1%
BAYESIAN TRIPHONE GMM-HMM [28]	25.6%
MONOPHONE HTMS [29]	24.8%
HETEROGENEOUS CLASSIFIERS [30]	24.4%
MONOPHONE RANDOMLY INITIALIZED DNNs (SIX LAYERS) [13]	23.4%
MONOPHONE DBN-DNNs (SIX LAYERS) [13]	22.4%
MONOPHONE DBN-DNNs WITH MMI TRAINING [31]	22.1%
TRIPHONE GMM-HMMs DT W/ BMMI [32]	21.7%
MONOPHONE DBN-DNNs ON FBANK (EIGHT LAYERS) [13]	20.7%
MONOPHONE MCRBM-DBN-DNNs ON FBANK (FIVE LAYERS) [33]	20.5%
MONOPHONE CONVOLUTIONAL DNNs ON FBANK (THREE LAYERS) [34]	20.0%

From: The paper

Experiments

- **Bing-Voice-Search Speech Recognition Task**
- This task used 24h of training data with a high degree of acoustic variability caused by noise, music, side-speech, accents, sloppy pronunciation, et al.
- The best DNN-HMM acoustic model achieved a sentence accuracy of 69.6% on the test set, compared with 63.8% for a strong, minimum phone error (MPE)-trained GMM-HMM baseline.

Experiments

- **Bing-Voice-Search Speech Recognition Task (Cont'd)**

[TABLE 2] COMPARING FIVE DIFFERENT DBN-DNN ACOUSTIC MODELS WITH TWO STRONG GMM-HMM BASELINE SYSTEMS THAT ARE DISCRIMINATIVELY TRAINED. SI TRAINING ON 309 H OF DATA AND SINGLE-PASS DECODING WERE USED FOR ALL MODELS EXCEPT FOR THE GMM-HMM SYSTEM SHOWN ON THE LAST ROW WHICH USED SA TRAINING WITH 2,000 H OF DATA AND MULTIPASS DECODING INCLUDING HYPOTHESES COMBINATION. IN THE TABLE, "40 MIX" MEANS A MIXTURE OF 40 GAUSSIANS PER HMM STATE AND "15.2 NZ" MEANS 15.2 MILLION, NONZERO WEIGHTS. WERs IN % ARE SHOWN FOR TWO SEPARATE TEST SETS, HUB500-SWB AND RT03S-FSH.

		WER	
MODELING TECHNIQUE	**#PARAMS [10^6]**	**HUB5'00-SWB**	**RT03S-FSH**
GMM, 40 MIX DT 309H SI	29.4	23.6	27.4
NN 1 HIDDEN-LAYER × 4,634 UNITS	43.6	26.0	29.4
+ 2 × 5 NEIGHBORING FRAMES	45.1	22.4	25.7
DBN-DNN 7 HIDDEN LAYERS × 2,048 UNITS	45.1	17.1	19.6
+ UPDATED STATE ALIGNMENT	45.1	16.4	18.6
+ SPARSIFICATION	15.2 NZ	16.1	18.5
GMM 72 MIX DT 2000H SA	102.4	17.1	18.6

From: The paper

Experiments

- **Other Large Vocabulary Tasks**
- Switchboard Speech Recognition Task (a corpus containing over 300h of training data)
- Google Voice Input Speech Recognition Task
- YouTube Speech Recognition Task
- English Broadcast News Speech Recognition Task

Experiments

- **Other Large Vocabulary Tasks (Cont'd)**

[TABLE 3] A COMPARISON OF THE PERCENTAGE WERs USING DNN-HMMs AND GMM-HMMs ON FIVE DIFFERENT LARGE VOCABULARY TASKS.

TASK	HOURS OF TRAINING DATA	DNN-HMM	GMM-HMM WITH SAME DATA	GMM-HMM WITH MORE DATA
SWITCHBOARD (TEST SET 1)	309	18.5	27.4	18.6 (2,000 H)
SWITCHBOARD (TEST SET 2)	309	16.1	23.6	17.1 (2,000 H)
ENGLISH BROADCAST NEWS	50	17.5	18.8	
BING VOICE SEARCH (SENTENCE ERROR RATES)	24	30.4	36.2	
GOOGLE VOICE INPUT	5,870	12.3		16.0 (>> 5,870 H)
YOUTUBE	1,400	47.6	52.3	

From: The paper

Content

- Speech Recognition System
- GMM-HMM Model
- Training Deep Neural Networks
- Generative Pretraining
- Experiments
- Discussion

Discussion

- **Convolutional DNNs for Phone Classification and Recognition**
- Although convolutional models along the temporal dimension achieved good classification results on TIMIT corpus, applying them to phone recognition is not straightforward.
- This is because temporal variations in speech can be partially handled by the dynamic programing procedure in the HMM component and hidden trajectory models.

Discussion

- **Speeding Up DNNs at Recognition Time**
- The time that a DNN-HMM system requires to recognize 1s of speech can be reduced from 1.6s to 210ms, without decreasing recognition accuracy, by quantizing the weights down to 8b using CPU.
- Alternatively, it can be reduced to 66ms by using a graphics processing unit (GPU).

Discussion

- **Alternative Pretraining Methods for DNNs**
- It is possible to learn a DNN by starting with a shallow neural net with a single hidden layer. Once this net has been trained discriminatively, a second hidden layer is interposed between the first hidden layer and the softmax output units and the whole network is again discriminatively trained. This can be continued until the desired number of hidden layers is reached, after which full backpropagation fine-tuning is applied.

Discussion

- **Alternative Pretraining Methods for DNNs (Cont'd)**
- Purely discriminative training of the whole DNN from random initial weights works well, too.
- Various types of autoencoder with one hidden layer can also be used in the layer-by-layer generative pretraining process.

Discussion

- **Alternative Fine-Tuning Methods for DNNs**
- Most DBN-DNN acoustic models are fine-tuned by applying stochastic gradient descent with momentum to small minibatches of training cases.
- More sophisticated optimization methods can be used, but it is not clear that the more sophisticated methods are worthwhile since the fine-tuning process is typically stopped early to prevent overfitting.

Discussion

- **Using DBN-DNNs to Provide Input Features for GMM-HMM Systems**
- This class of methods use neural networks to provide the feature vectors for the training process of the GMM in a GMM-HMM system.
- The most common approach is to train a randomly initialized neural net with a narrow bottleneck middle layer and to use the activations of the bottleneck hidden units as features.

Discussion

- **Using DNNs to Estimate Articulatory Features for Detection-Based Speech Recognition**
- DBN-DNNs are effective for detecting subphonetic speech attributes (also known as phonological or articulatory features).

Discussion

- **Summary**
- Most of the gain comes from using DNNs to exploit information in neighboring frames and from modeling tied context-dependent states.
- There is no reason to believe that the optimal types of hidden units or the optimal network architectures are used, and it is highly likely that both the pretraining and fine-tuning algorithms can be modified to reduce the amount of overfitting and the amount of computation.

Thank You!

Investigation of Speech Separation as a Front-End for Noise Robust Speech Recognition

Presented by Peidong Wang

04/04/2016

Narayanan, Arun, and DeLiang Wang. "Investigation of speech separation as a front-end for noise robust speech recognition." *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* 22.4 (2014): 826-835.

Content

- Introduction
- System Description
- Evaluation Results
- Discussion

Content

- Introduction
- System Description
- Evaluation Results
- Discussion

Introduction

- **Background**
- Although automatic speech recognition (ASR) systems have become fairly powerful, the inherent variability can still pose challenges.
- Typically, ASR systems that work well in clean conditions suffer from a drastic loss of performance in the presence of noise.

Introduction

- **Feature-Based Methods**
- This class of methods focus on feature extraction or feature normalization.
- Feature-based techniques have the potential to generalize well, but do not always produce the best results.

Introduction

- **Two Groups of Feature-Based Methods**
- When stereo [*] data is unavailable, prior knowledge about speech and/or noise is used, such as spectral reconstruction based missing feature methods, direct masking methods and feature enhancement methods.
- When stereo data is available, feature mapping methods and recurrent neural networks have been used.

[*] By stereo we mean noisy and the corresponding clean signals.

Introduction

- **Model-Based Methods**
- The ASR model parameters are adapted to match the distribution of noisy or enhanced features.
- Model-based methods work well when the underlying assumptions are met, but typically involve significant computational overhead.
- The best performances are usually obtained by combining feature-based and model-based methods.

Introduction

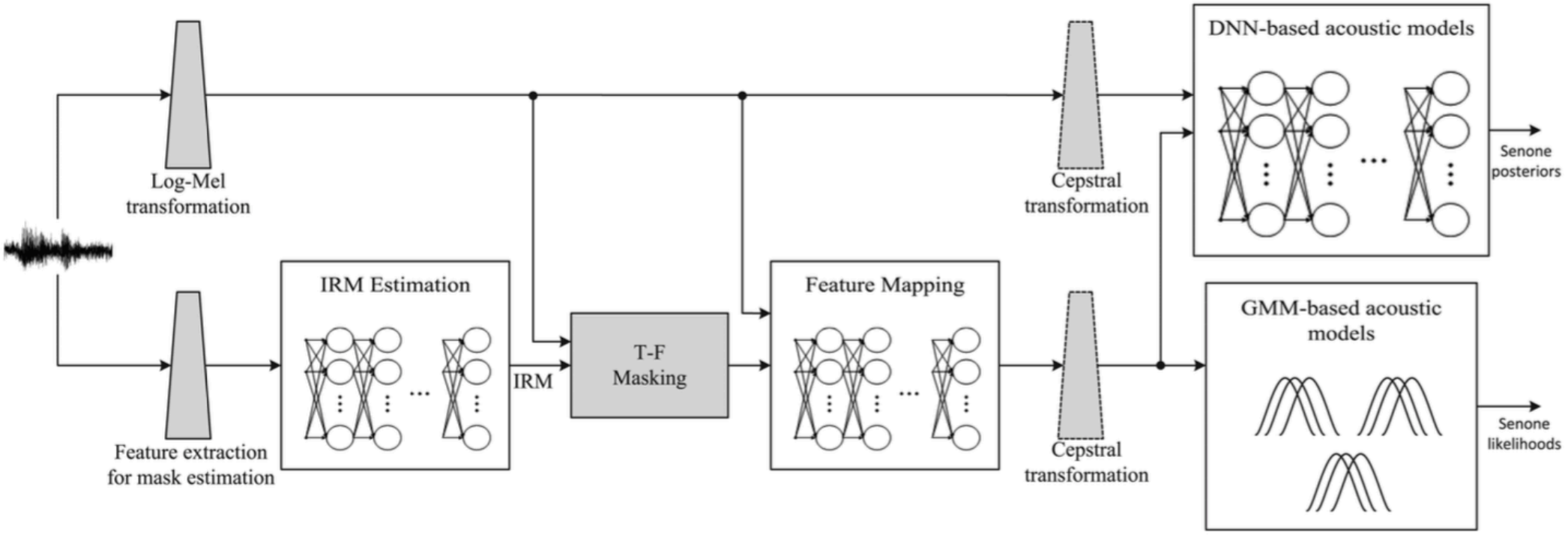
- **Supervised Classification Based Speech Separation**
- Stereo training data is also used by supervised classification based speech separation algorithms.
- Such algorithms typically estimate the ideal binary mask (IBM)-a binary mask defined in the time-frequency (T-F) domain that identifies speech dominant and noise dominant T-F units.
- The above method can be extended to ideal ratio mask (IRM), which represents the ratio of speech to mixture energy.

Content

- Introduction
- System Description
- Evaluation Results
- Discussion

System Description

- **Block Diagram of the Proposed System**



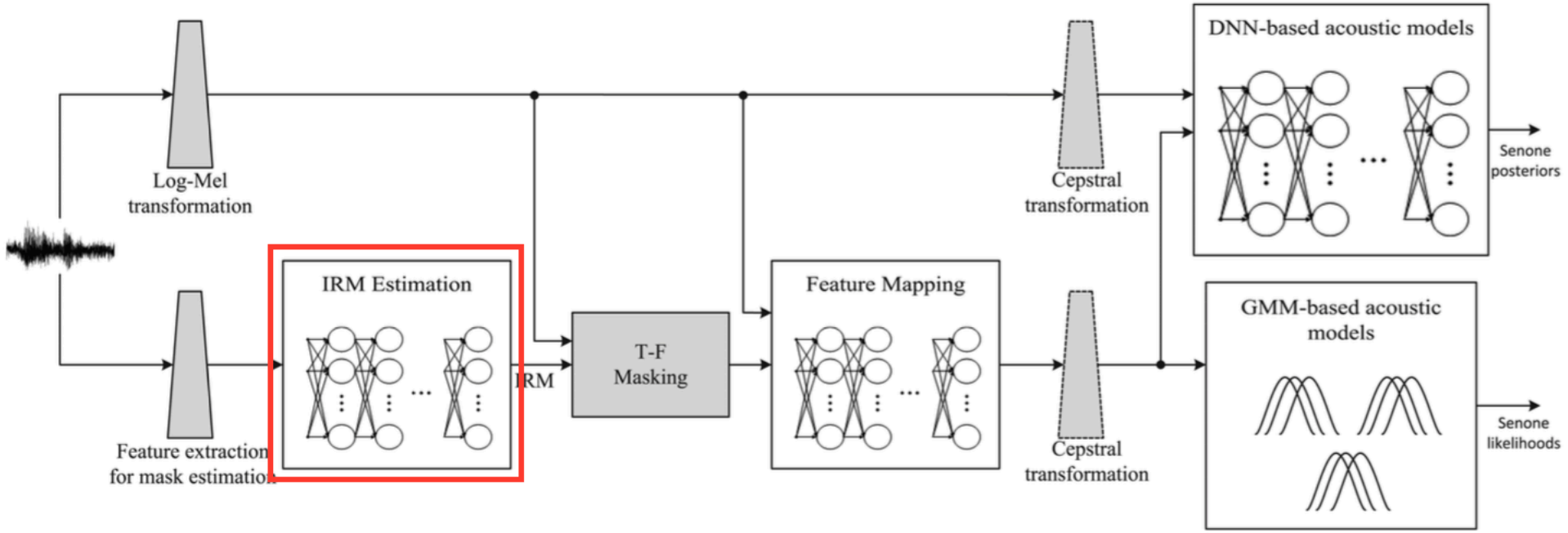
From: The paper

System Description

- **Addressing Additive Noise and Convolutional Distortion**
- The additive noise and the convolutional distortion are dealt with in two separate stages: Noise removal followed by channel compensation.
- Noise is removed via T-F masking using the IRM. To compensate for channel mismatch and the errors introduced by masking, we learn a non-linear mapping function that undoes these distortions.

System Description

- **Time-Frequency Masking**



System Description

- **Time-Frequency Masking (Cont'd)**
- Here the authors perform T-F masking in the mel-frequency domain, unlike some of the other systems that operate in the gammatone feature domain.
- To obtain the mel-spectrogram of a signal, it is first pre-emphasized and transformed to the linear frequency domain using a 320 channel fast Fourier transform (FFT). A 20msec Hamming window is used. The 161-dimensional spectrogram is then converted to a 26-channel mel-spectrogram.

System Description

- **Time-Frequency Masking (Cont'd)**
- The authors use DNNs to estimate the IRM as DNNs show good performance and training using stochastic gradient descent scales well compared to other nonlinear discriminative classifiers.

System Description

- **Time-Frequency Masking (Cont'd)**
- Target Signal
- The ideal ratio mask is defined as the ratio of the clean signal energy to the mixture energy at each time-frequency unit.
- The mathematical expression is shown below.

$$IRM(t,f) = \frac{10^{(SNR(t,f)/10)}}{10^{(SNR(t,f)/10)} + 1}$$

$$SNR(t,f) = 10\log_{10}(X(t,f)/N(t,f))$$

System Description

- **Time-Frequency Masking (Cont'd)**
- Target Signal
- Rather than estimating IRM directly, the authors estimate a transformed version of the SNR.
- The mathematical expression of the sigmoidal transformation is shown below.

$$d(t,f) = \frac{1}{1+\exp(-\alpha(SNR(t,f)-\beta))}$$

System Description

- **Time-Frequency Masking (Cont'd)**
- Target Signal
- During testing, the values output from the DNN are mapped back to their corresponding IRM values.

System Description

- **Time-Frequency Masking (Cont'd)**
- Features
- Feature extraction is performed both at the fullband and the subband level.
- The combination of features, 31 dimensional MFCCs, 13 dimensional FASTA filtered PLPs and 15 dimensional amplitude modulation spectrogram (AMS) features, are used.

System Description

- **Time-Frequency Masking (Cont'd)**
- Features
- The fullband features are derived by splicing together fullband MFCCs and RASTA-PLPs, along with their delta and acceleration components, and subband AMS features.
- The subband features are derived by splicing together subband MFCCs, RASTA-PLPs, and AMS features. Some auxiliary components are also added.

System Description

- **Time-Frequency Masking (Cont'd)**
- Supervised Learning
- IRM estimation is performed in two stages. In the first stage, multiple DNNs are trained using fullband and subband features. The final estimate is obtained using an MLP that combines the output of the fullband and the subband DNNs.

System Description

- **Time-Frequency Masking (Cont'd)**
- Supervised Learning
- The fullband DNNs would be cognizant of the overall spectral shape of the IRM and the information conveyed by the fullband features, whereas the subband DNNs are expected to be more robust to noise occurring at frequencies outside their passband.

System Description

- **Time-Frequency Masking (Cont'd)**

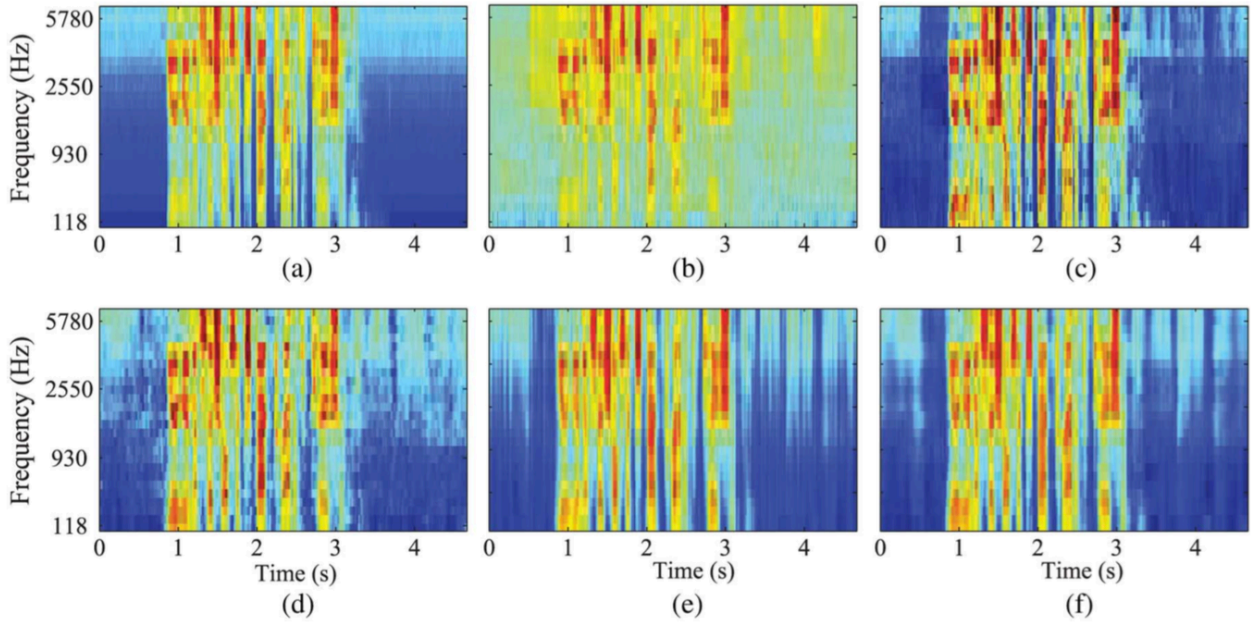
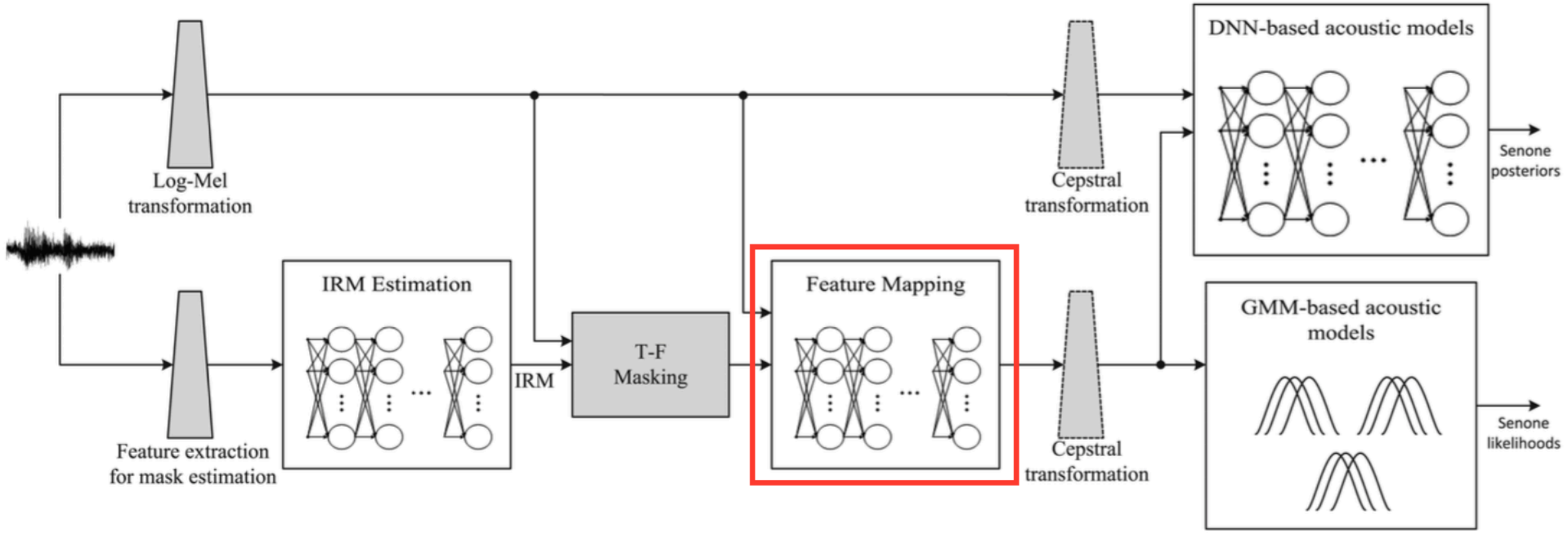


Fig. 2. (Color online) Example of T-F masking. (a) Mel spectrogram of a clean signal from the Aurora-4 corpus. (b) Mel spectrogram of the same signal mixed with babble noise. (c) Map of the true instantaneous SNRs, expressed in dB. (d) Instantaneous SNRs estimated using subband features and the corresponding DNNs. The mean absolute SNR estimation error for this mask is 2.9 dB. (e) Instantaneous SNRs estimated using fullband features. SNR estimation error for this mask is 2.5 dB. (e) Instantaneous SNR estimates obtained after combining the masks in (d) and (e). It can be noticed that the mask in (f) is smoother than those in (d) and (e). SNR estimation error for this mask is 2.1 dB. Note that the SNR estimates are rounded to the range $[-15, 10]$ dB before calculating the mean absolute error.

From: The paper

System Description

- **Feature Mapping**



System Description

- **Feature Mapping (Cont'd)**
- Even after T-F masking, channel mismatch can still significantly impact performance.
- This happens for two reasons. Firstly, the algorithm learns to estimate the ratio mask using mixtures of speech and noise recorded using a single microphone. Secondly, because channel mismatch is convolutional, speech and noise, which now includes both background noise and convolutive noise, are clearly not uncorrelated.

System Description

- **Feature Mapping (Cont'd)**
- The goal of feature mapping in this work is to learn spectro-temporal correlations that exist in speech to undo the distortions introduced by unseen microphones and the first stage of the algorithm.

System Description

- **Feature Mapping (Cont'd)**
- Target Signal
- The target is the clean log-mel spectrogram (LMS). The "clean" LMS here corresponds to those obtained from the clean signals recorded using a single microphone in a single filter setting.

System Description

- **Feature Mapping (Cont'd)**
- Target Signal
- Instead of using the LMS directly as the target, the authors apply a linear transform to limit the target values to the range [0, 1] to use the sigmoidal transfer function for the output layer of the DNN.
- The mathematical expression is as follows.

$$X_d(t,f) = \frac{\ln(X(t,f)) - \min(\ln(X(\cdot,f)))}{\max(\ln(X(\cdot,f))) - \min(\ln(X(\cdot,f)))}$$

System Description

- **Feature Mapping (Cont'd)**
- Target Signal
- During testing, the output of the DNN is mapped back to the dynamic range of the utterances in training set.

System Description

- **Feature Mapping (Cont'd)**
- Features
- The authors use both the noisy and the masked LMS.

- Supervised Learning
- Unlike the DNNs used for IRM estimation, the hidden layers of the DNN for this task use rectified linear units (ReLUs). In addition, the output layer uses sigmoid activations.

System Description

- **Feature Mapping (Cont'd)**

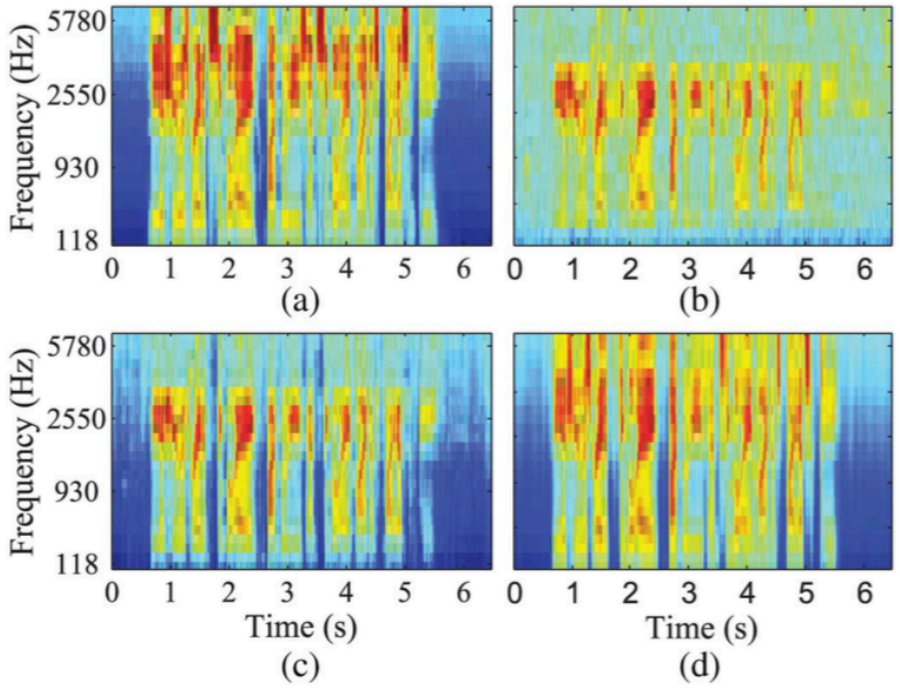
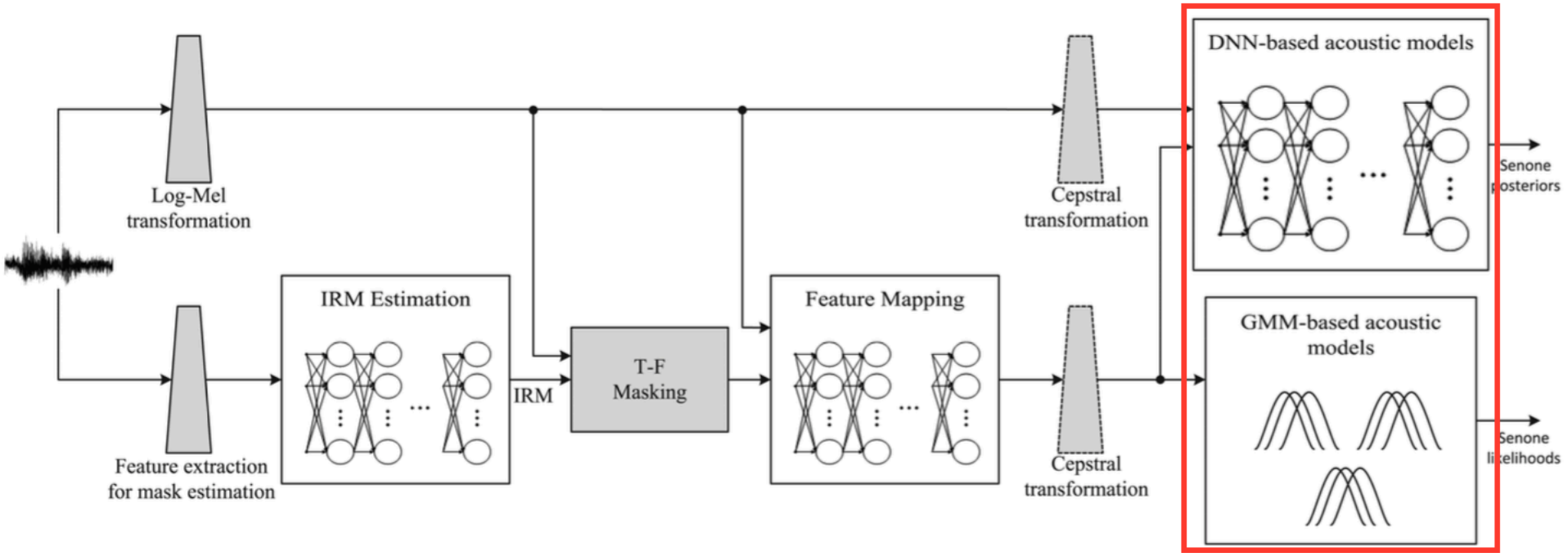


Fig. 3. (Color online) Example of feature mapping. (a) Mel spectrogram of the clean signal recorded using a Sennheiser microphone and processed with P.314 filter. (b) Mel spectrogram of the signal recorded using an alternative microphone and mixed with babble noise. The microphone attenuates the high frequency components of the signal. (c) The mel spectrogram after T-F masking. Noise has largely been removed but the high frequency components are still attenuated. (d) The mel spectrogram after feature mapping. As can be seen, the high frequency components are reconstructed reasonably well.

From: The paper

System Description

- **Acoustic Modeling**



System Description

- **Acoustic Modeling (Cont'd)**
- The acoustic models are trained using the Aurora-4 dataset.
- Aurora-4 is a 5000-word closed vocabulary recognition task based on the Wall Street Journal database. The corpus has two training sets, clean and multi-condition, both with 7138 utterances.

System Description

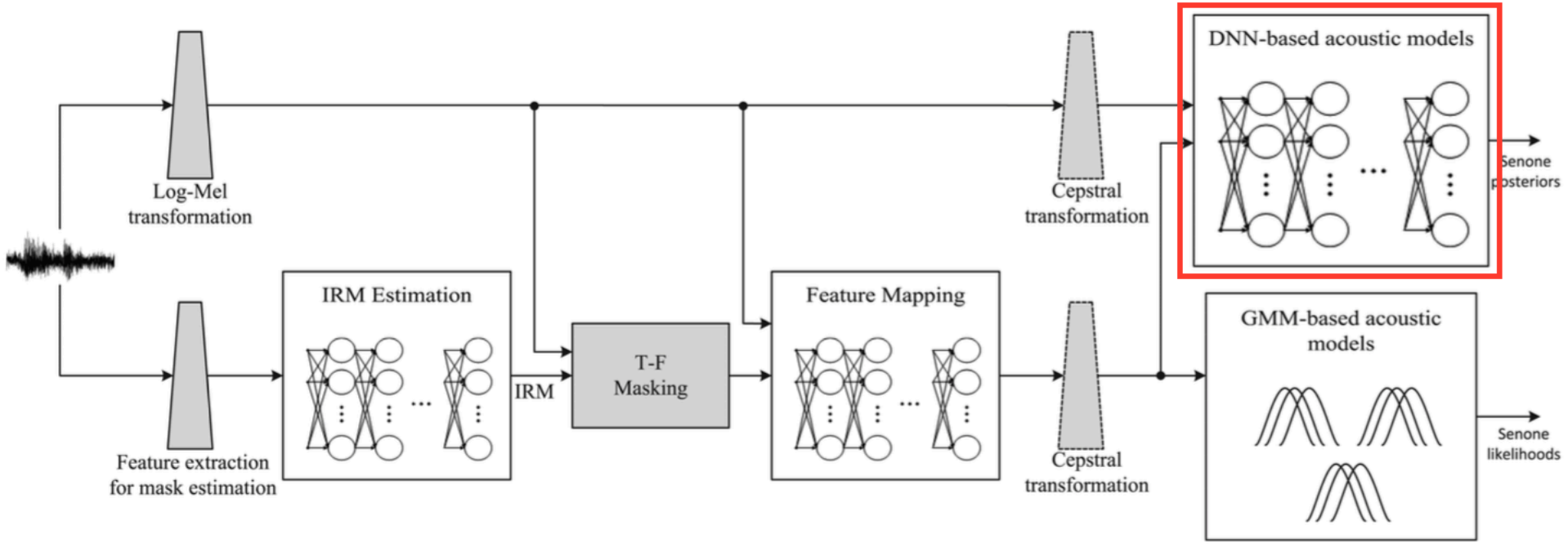
- **Acoustic Modeling (Cont'd)**
- Gaussian Mixture Models
- The HMMs and the GMMs are initially trained using the clean training set. The clean models are then used to initialize the multi-condition models; both clean and multi-condition models have the same structure and differ only in transition and observation probability densities.

System Description

- **Acoustic Modeling (Cont'd)**
- Deep Neural Networks
- The authors first align the clean training set to obtain senone labels at each time-frame for all utterances in the training set. DNNs are then trained to predict the posterior probability of senones using either clean features or features extracted from the multi-condition set.

System Description

- **Diagonal Feature Discriminant Linear Regression**



System Description

- **Diagonal Feature Discriminant Linear Regression (Cont'd)**
- dFDLR is a semi-supervised feature adaptation technique.
- The motivation for developing dFDLR is to address the problem of generalization to unseen microphone conditions in our dataset, which is where the DNN-HMM systems perform the worst.

System Description

- **Diagonal Feature Discriminant Linear Regression (Cont'd)**
- To apply dFDLR, we first obtain an initial senone-level labeling for our test utterances using the unadapted models. Features are then transformed to minimize the cross-entropy error in predicting these labels.
- The mathematical expressions are as follow.

$$\hat{O}_t(f) = w_f \bullet O_t(f) + b_f$$

$$\min \sum_t E(s_t, D_{out}(\hat{O}_{t-5} \ldots \hat{O}_{t+5}))$$

System Description

- **Diagonal Feature Discriminant Linear Regression (Cont'd)**
- The parameters can easily be learned within the DNN framework by adding a layer between the input layer and the first hidden layer of the original DNN. After initialization, the standard backpropagation algorithm is run for 10 epochs to learn the parameters of the dFDLR model. During backpropagation, weights of the original hidden layers are kept unchanged and only the parameters in the dFDLR are updated.

Content

- Introduction
- System Description
- Evaluation Results
- Discussion

Evaluation Results

TABLE I

WORD ERROR RATES ON THE AURORA-4 CORPUS USING THE GMM-HMM SYSTEMS. THE COLUMNS CLEAN, NOISY, CLEAN + CHANNEL, AND NOISY + CHANNEL CORRESPOND TO THE WER AVERAGED ON TEST SETS 1, 2 TO 7, 8, AND 9 TO 14, RESPECTIVELY. THE BEST PERFORMANCE IN EACH CONDITION IS MARKED IN BOLD. RESULTS OBTAINED USING VTS-BASED MODEL ADAPTATION IS ALSO SHOWN

System	Clean	Noisy	Clean + Channel	Noisy + Channel	Average
Clean Training					
Noisy	9.1	27.0	22.9	44.3	32.8
AFE	**9.0**	23.2	29.9	38.4	29.2
Feature mapping	9.8	16.3	14.3	29.6	21.4
T-F masking	9.4	15.3	23.1	36.4	24.5
+ feature mapping	9.7	**15.2**	**14.1**	**28.9**	**20.6**
Multi-condition Training					
Noisy	10.6	17.2	17.7	31.8	23.0
AFE	**10.3**	18.4	20.0	30.4	23.1
Feature mapping	11.7	15.9	14.7	27.5	20.5
T-F masking	10.7	**14.3**	20.1	31.7	21.9
+ feature mapping	11.8	16.0	**14.5**	**27.2**	**20.4**
VTS [45]	6.9	15.1	11.8	23.3	17.8

From: The paper

Evaluation Results

TABLE II
WORD ERROR RATES ON THE AURORA-4 CORPUS USING THE DNN-HMM SYSTEMS TRAINED IN CLEAN CONDITIONS. THE BEST PERFORMANCE IN EACH CONDITION IS MARKED IN BOLD

System	Clean	Noisy	Clean + Channel	Noisy + Channel	Average
Cepstral features					
Noisy	5.3	19.2	18.6	36.1	25.4
+ dFDLR	**5.2**	19.6	17.5	35.9	25.4
Feature mapping	5.4	9.9	9.5	22.9	15.1
+ dFDLR	5.4	9.7	8.8	22.3	14.7
Proposed frontend	5.6	9.3	9.5	22.0	14.5
+ dFDLR	5.4	**9.2**	**9.1**	**21.4**	**14.1**
log-mel features					
Noisy	5.2	22.9	21.3	41.6	29.5
+ dFDLR	5.1	22.7	20.5	41.4	29.3
Feature mapping	5.3	10.7	9.6	25.2	16.4
+ dFDLR	5.3	10.3	**8.7**	24.5	15.9
Proposed frontend	5.2	9.5	9.6	24.0	15.4
+ dFDLR	**4.9**	**9.2**	9.0	**23.3**	**14.9**

TABLE III
WORD ERROR RATES ON THE AURORA-4 CORPUS USING THE DNN-HMM SYSTEMS TRAINED ON THE MULTI-CONDITION SET. THE BEST PERFORMANCE IN EACH CONDITION IS MARKED IN BOLD

System	Clean	Noisy	Clean + Channel	Noisy + Channel	Average
Cepstral features					
Noisy	6.7	10.9	10.4	21.9	15.3
+ dFDLR	6.7	11.0	9.8	21.4	15.1
Proposed frontend	6.7	10.0	10.2	20.3	14.2
+ dFLDR	6.8	9.7	9.5	19.7	13.7
Concat-features	5.6	8.9	9.4	20.5	13.6
+ dFDLR	**5.4**	**8.7**	**8.8**	**20.0**	**13.3**
log-mel features					
Noisy	5.3	8.5	9.0	18.2	12.5
+ dFDLR	5.1	8.5	8.4	**17.6**	**12.1**
Proposed frontend	5.3	9.0	8.9	21.9	14.3
+ dFDLR	5.3	8.8	8.7	21.3	13.9
Concat-features	4.9	8.4	8.3	20.4	13.3
+ dFDLR	**4.8**	**8.2**	**8.1**	20.0	13.0

From: The paper

Content

- Introduction
- System Description
- Evaluation Results
- Discussion

Discussion

- Several interesting observations can be made from the results presented in the previous section.
- Firstly, the results clearly show that the speech separation front-end is doing a good job at removing noise and handling channel mismatch.
- Secondly, with no channel mismatch, T-F masking alone worked well in removing noise.

Discussion

- Finally, directly performing feature mapping from noisy features to clean features performs reasonably, but it does not perform as well as the proposed front-end.

Thank You!