4. Please describe the relationship between GMM-HMM and DNN-HMM based speech recognition systems. How to calculate P(X|W) by DNN?

First of all, both the GMM-HMM and DNN-HMM are used as the acoustic model in the speech recognition system. They play the same role in determining P(X|W) (the probability of the feature vectors given the recognized words).

Plus, both of them use HMM to make a transition from its current state (phoneme) to one of its connected states every time step.

One of the differences is that the observation probabilities are generated from GMM or DNN. In GMM-HMM, every state has a probability distribution described by the GMM. In contrast, DNN-HMM estimates the posterior probability of each state from the sequence of acoustic feature.

Another main difference is that GMM-HMM has its limitations in modeling the continuous-time speech signal, while DNN-HMM can easily model correlated features. The reason is that unlike GMM-HMM uses the feature from one single frame as input, DNN-HMM is not built on the hypothesis that the acoustic features from different frames are independent of each other, thus can utilizing the context information from multiple continuous frames.

At last, both the GMM-HMM and DNN-HMM may share the same decoding algorithm, like Viterbi algorithm.

When it comes to how to calculate P(X|W) by DNN, first we need to generate labels using a trained GMM-HMM, then train the DNN to associate a phone label with a frame of acoustic feature using the cross-entropy loss function to optimize. Finally, the DNN outputs the posterior P(W|X), then Bayes Rule can be applied here to calculate the P(X|W): $P(X|W)=\frac{P(W|X)P(X)}{P(W)}$.

5. Please summarize the advantages and disadvantages of DNN-HMM and end-to-end ASR system, and explain the main solutions to address these disadvantages of end-to-end ASR system.

Advantages of DNN-HMM:

1) The DNN-HMM can utilize the context information of frames by estimateing the posterior probability of each state from the sequence of acoustic feature and is stable processing to estimate phoneme states in a frame-by-frame manner.

2) The training can be performed using the Viterbi algorithm and the decoding is generally quite efficient.

Disadvantages of DNN-HMM:

1) The training process is complex and difficult to be globally optimized. HMM-based model often uses different training methods and data sets to train different modules. Each module is independently optimized with their own optimization objective functions which are generally different from the true LVCSR performance evaluation criteria. So the optimality of each module does not necessarily mean the global optimality. That being said, the cascading processing of DNN-HMM may lead to omitting global optimization and get local optimization instead.

2) Conditional independent assumptions. The HMM-based model uses conditional independence assumptions within HMM and between different modules. This does not match the actual situation of LVCSR.

Advantages of end-to-end ASR system:

1) Compared with the HMM-based model, the end-to-end model uses a single model to directly map audio to characters or words with no requirement of domain expertise, thus is simpler for constructing and training.

2) End-to-end ASR system can achieve the total optimization.

Disadvantages of end-to-end ASR system:

1) End-to-end ASR system often suffers from the problem in which redundant generations repeat and importance symbols vanish.

2) Current performance of the end-to-end model is still worse than that of the HMM-DNN model, at best just comparable.

Looking ahead, the end-to-end ASR system needs to at least be improved in the following aspects:

1) Better trade-off the model delay and the recognition performance. Reducing latency while ensuring the recognition accuracy is an important but challenging research issue for the end-to-end model.

2) Better language knowledge learning. HMM-based model uses additional language models to provide a wealth of language knowledge, while the end-to-end model can only learn from limited training data’s transcriptions. This leads to great difficulties in dealing with scenes with large linguistic diversity.

6. Please explain the main problems and solutions of traditional signal processing based and deep learning based single channel speech enhancement approaches.

Both of the traditional signal processing based and deep learning based single channel speech enhancement approaches have a common problem: the estimation of the noise signal.

Traditional signal processing based single channel speech enhancement approaches

1) Spectral Subtraction

In Spectral Subtraction, noise signals are assumed to be additive, so the estimate of the underlying clean speech spectrum could be obtained by subtracting the estimate of noise spectrum from the noisy spectrum. The noise spectrum is estimated from the silent periods i.e., absence of the speech signals.

2) Wiener Filtering

Wiener filtering based speech enhancement minimizes the mean square error (MSE) between the estimated speech magnitude spectrum and the original signal magnitude spectrum.

The traditional speech enhancement methods mentioned above work well when the additional noise signal is stationary. However, the hypotheses for such algorithms do not well under the non-stationary noisy conditions. This is the time we need to utilize the non-linear modeling ability of deep learning based speech enhancement methods to get better enhanced results.

The deep learning based single channel speech enhancement approaches have another unique problem: its low generalization.

Most spectral domain based speech enhancement techniques exploit some higher-level feature. Recent advanced speech enhancement approaches mainly operate on the waveform of signal directly and further improved speech quality. However, generalization remains a major problem in deep learning based single channel speech enhancement. In [1], they propose *learnable loss mixup (LLM)*, a simple and effort-less training diagram, to improve the generalization of deep learning-based speech enhancement models.

[1] Single-channel speech enhancement using learnable loss mixup

7. What is the difference between generative embedding with DNN i-vector and deep speaker embeddings? How to extract X-vector embeddings?

We use a low-dimensional “identity vector” (i-vector for short) to represent a speech segment. An i-vector contains the voice characteristic of a person (attributed to the speaker subspace) and channel factors (attributed to the channel subspace). The i-vector approach has become state-of-the-art in the speaker verification field. The approach provides an elegant way of reducing high-dimensional sequential input data to a low-dimensional fixed-length feature vector while retaining most of the relevant information. In general, we may extract the i-vector by the following steps:
1) Extract from waveform to get acoustic feature vectors
2) Use a universal background model (which is a GMM) to extract sufficient statistics
3) Obtain i-vector using a low-rank projection
4) Score with PLDA
The architecture for deep speaker embedding is an encode-decoder model. Encoder takes a sequence of acoustic features and derive intermediate representations. Temporal aggregation converts the sequence of intermediate representations into a single fixed-dimensional vector. In decoder, one of the layers is designed to be a bottleneck layer whose output (before the non-linearity) is taken as the speaker embedding. The encoder-decoder network is trained end-to-end to classified utterances from a large set of speakers.

The x-vector system is based on the DNN i-vector architecture and its the configuration is outlined in the table below. x-vectors are extracted at layer *segment6*, before the nonlinearity.

Layer	Layer context	Total context	Input x output
frame1	[t-2,t+2]	5	120x512
frame2	{t-2,t,t+2}	9	1536x512
frame3	{t-3,t,t+3}	15	1536x512
frame4	{t}	15	512x512
frame5	{t}	15	512x1500
stats pooling	[0,T)	*T*	1500*T*x3000
segment6	{0}	*T*	3000x512
segment7	{0}	*T*	512x512
softmax	{0}	*T*	512x*N*

Suppose an input segment has *T* frames. The first five layers operate on speech frames, with a small temporal context centered at the current frame *t*. The statistics pooling layer aggregates all *T* frame-level outputs from layer *frame5* and computes its mean and standard deviation. The statistics are 1500 dimensional vectors, computed once for each input segment. This process aggregates information across the time dimension so that subsequent layers operate on the entire segment. The mean and standard deviation are concatenated together and propagated through segment-level layers and finally the softmax output layer.