Who Can Understand Your Speech Better -- Deep Neural Network or Gaussian Mixture Model?

Dong Yu
Microsoft Research

Thanks to my collaborators:
Li Deng, Frank Seide, Gang Li, Mike Seltzer, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Adam Eversole, George Dahl, Abdel-rahman Mohamed, Xie Chen, Hang Su, Ossama Abdel-Hamid, Eric Wang, Andrew Maas, and many more
Demo: Real Time Speech to Speech Translation

http://youtu.be/Nu-nlQqFCKg

Microsoft Chief Research Officer Dr. Rick Rashid demoed the real time speech-to-speech translation technique at 14th Computing in the 21st Century Conference held at Tianjin, China, on Oct. 25, 2012.
Speech to Speech Translation

Speech Recognition
Frank Seide
Gang Li
Dong Yu
Li Deng

Machine Translation
Xiaodong He
Dongdong Zhang
Mei-Yuh Hwang
Mu Li
Mohamed Abdel-Hady
Ming Zhou

Personalized Speech Synthesis
Yao Qian
Frank Soong
Lijuan Wang

Project Management: Noelle Sophy, Chris Wendt
Speech to Speech Translation

SI DNN trained with 2000-hr SWB data
Has 180 million parameters

DNN

32k tied triphone states

Observation Probabilities

Transition Probabilities

7 hidden layers each with 2048 neurons

11 frames of 52-dim plp feature

12/7/2012
Dong Yu: Keynote at IWSLT 2012
DNN-HMM Performs Very Well

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

<table>
<thead>
<tr>
<th>AM</th>
<th>Setup</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM-HMM</td>
<td>MPE (760 24-mixture)</td>
<td>36.2%</td>
</tr>
<tr>
<td>DNN-HMM</td>
<td>5 layers x 2048</td>
<td>30.1% (-17%)</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>AM</th>
<th>Setup</th>
<th>Hub5’00-SWB</th>
<th>RT03S-FSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM-HMM</td>
<td>BMMI (9K 40-mixture)</td>
<td>23.6%</td>
<td>27.4%</td>
</tr>
<tr>
<td>DNN-HMM</td>
<td>7 x 2048</td>
<td>15.8% (-33%)</td>
<td>18.5% (-33%)</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>AM</th>
<th>Setup</th>
<th>Hub5’00-SWB</th>
<th>RT03S-FSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM-HMM (A)</td>
<td>BMMI (18K 72-mixture)</td>
<td>21.7%</td>
<td>23.0%</td>
</tr>
<tr>
<td>GMM-HMM (B)</td>
<td>BMMI + fMPE</td>
<td>19.6%</td>
<td>20.5%</td>
</tr>
<tr>
<td>DNN-HMM</td>
<td>7 x 3076</td>
<td>14.4% (A: -34% B: -27%)</td>
<td>15.6% (A: -32% B: -24%)</td>
</tr>
</tbody>
</table>
DNN-HMM Performs Very Well

- **Microsoft** audio video indexing service (Knies, 2012)
  - “It’s a big deal. The benefits, says Behrooz Chitsaz, director of Intellectual Property Strategy for Microsoft Research, are improved accuracy and faster processor timing. He says that 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, a technology increasingly important to Google's smartphone operating system, Android, as well as the search app it makes available for Apple devices (see "Google's Answer to Siri Thinks Ahead"). "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. "That means that many more people will have a perfect experience without errors.”
Outline

- CD-DNN-HMM
- Invariant Features
- Once Considered Obstacles
- Other Advances
- 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.

\[
p(l = k | h; \theta) = \frac{e^{x \sum_{i=1}^{H} \lambda_{ik} h_i + a_k}}{Z(h)}
\]

- 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.

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

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

- 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 $h_0$ as the data
  - Continue the process to train the next layer
- Intuitively $\log p(\nu)$ 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 $h_0$ as the data
  - Continue the process to train the next layer
- Intuitively $\log p(v)$ 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 $h_0$ as the data
  - Continue the process to train the next layer

- Intuitively $\log p(\nu)$ 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 $h_0$ as the data
  - Continue the process to train the next layer
- Intuitively $\log p(v)$ improves as new layer is added and trained.
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)

<table>
<thead>
<tr>
<th>Invariant Features</th>
<th>Other Advances</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model senones (tied triphone states) directly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Many layers of nonlinear feature transformation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Long window of frames</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Modeling Senones is Critical

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

<table>
<thead>
<tr>
<th>Model</th>
<th>monophone</th>
<th>senone</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM-HMM MPE</td>
<td>-</td>
<td>36.2</td>
</tr>
<tr>
<td>DNN-HMM 1× 2K</td>
<td>41.7</td>
<td>31.9</td>
</tr>
<tr>
<td>DNN-HMM 3×2K</td>
<td>35.8</td>
<td>30.4</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Model</th>
<th>monophone</th>
<th>senone</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM-HMM BMMI</td>
<td>-</td>
<td>23.6</td>
</tr>
<tr>
<td>DNN-HMM 7×2K</td>
<td>34.9</td>
<td>17.1</td>
</tr>
</tbody>
</table>

- 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%)

<table>
<thead>
<tr>
<th>Model</th>
<th>1 frame</th>
<th>11 frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD-DNN-HMM 1×4634</td>
<td>26.0</td>
<td>22.4</td>
</tr>
<tr>
<td>CD-DNN-HMM 7×2k</td>
<td>23.2</td>
<td>17.1</td>
</tr>
</tbody>
</table>

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 sequential criterion 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%)

<table>
<thead>
<tr>
<th>L×N</th>
<th>DBN-Pretrain</th>
<th>1×N</th>
<th>DBN-Pretrain</th>
</tr>
</thead>
<tbody>
<tr>
<td>1×2k</td>
<td>24.2</td>
<td>1×2k</td>
<td>24.2</td>
</tr>
<tr>
<td>2×2k</td>
<td>20.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3×2k</td>
<td>18.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4×2k</td>
<td>17.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5×2k</td>
<td>17.2</td>
<td>1×3772</td>
<td>22.5</td>
</tr>
<tr>
<td>7×2k</td>
<td>17.1</td>
<td>1×4634</td>
<td>22.6</td>
</tr>
<tr>
<td>9×2k</td>
<td>17.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9×1k</td>
<td>17.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5×3k</td>
<td>17.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>1×16k</strong></td>
<td><strong>22.1</strong></td>
</tr>
</tbody>
</table>
### Pretraining Helps but Not Critical
(Seide, Li, Yu 2011, Seide, Li, Chen, Yu 2011)

<table>
<thead>
<tr>
<th>$L \times N$</th>
<th>DBN-Pretrain</th>
<th>BP</th>
<th>LBP</th>
<th>Discriminative Pretrain</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \times 2k$</td>
<td>24.2</td>
<td>24.3</td>
<td>24.3</td>
<td>24.1</td>
</tr>
<tr>
<td>$2 \times 2k$</td>
<td>20.4</td>
<td>22.2</td>
<td>20.7</td>
<td>20.4</td>
</tr>
<tr>
<td>$3 \times 2k$</td>
<td>18.4</td>
<td>20.0</td>
<td>18.9</td>
<td>18.6</td>
</tr>
<tr>
<td>$4 \times 2k$</td>
<td>17.8</td>
<td>18.7</td>
<td>17.8</td>
<td>17.8</td>
</tr>
<tr>
<td>$5 \times 2k$</td>
<td>17.2</td>
<td>18.2</td>
<td>17.4</td>
<td>17.1</td>
</tr>
<tr>
<td>$7 \times 2k$</td>
<td>17.1</td>
<td>17.4</td>
<td>17.4</td>
<td>16.8</td>
</tr>
<tr>
<td>$9 \times 2k$</td>
<td>17.0</td>
<td>16.9</td>
<td>16.9</td>
<td>-</td>
</tr>
<tr>
<td>$9 \times 1k$</td>
<td>17.9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$5 \times 3k$</td>
<td>17.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

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

- CD-DNN-HMM
- Invariant Features
- Once Considered Obstacles
- Other Advances
- Summary
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
What Makes ASR Difficult?

Variability, Variability, Variability

<table>
<thead>
<tr>
<th>Speaker</th>
<th>Environment</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Accents</td>
<td>• Noise</td>
<td>• Head phone</td>
</tr>
<tr>
<td>• Dialect</td>
<td>• Side talk</td>
<td>• Land phone</td>
</tr>
<tr>
<td>• Style</td>
<td>• Reverberation</td>
<td>• Speaker phone</td>
</tr>
<tr>
<td>• Emotion</td>
<td>•</td>
<td>• Cell phone</td>
</tr>
<tr>
<td>• Coarticulation</td>
<td>•</td>
<td></td>
</tr>
<tr>
<td>• Reduction</td>
<td>•</td>
<td></td>
</tr>
<tr>
<td>• Pronunciation</td>
<td>•</td>
<td></td>
</tr>
<tr>
<td>• Hesitation</td>
<td>•</td>
<td></td>
</tr>
</tbody>
</table>

Interactions between these factors are complicated and nonlinear
DNN Learns Invariant and Discriminative Features

- **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

![Diagram](image)

Log-linear classifier

Many layers of nonlinear feature transformation

CD-DNN-HMM | Invariant Features | Once Considered Obstacles | Other Advances | Summary

CD-DNN-HMM | Invariant Features | Once Considered Obstacles | Other Advances | Summary

12/7/2012
Dong Yu: Keynote at IWSLT 2012
Higher Layer Features More Invariant

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

\[ \| \delta^{l+1} \| = \| \sigma(z^l(v^l + \delta^l)) - \sigma(z^l(v^l)) \| \approx \| \text{diag}(\sigma'(z^l(v^l(t)))) (w^l)^T \delta^l \| \leq \| \text{diag}(\sigma'(z^l(v^l(t)))) (w^l)^T \| \| \delta^l \| \]

Percentage of weights whose magnitude is below the threshold

- Layer 1
- Layer 2
- Layer 3
- Layer 4
- Layer 5
- Layer 6
- Layer 7

Weight magnitude

1% 5% 9% 13% 17% 21% 25% 29% 33% 37% 41% 45% 49% 53% 57% 61% 65% 69% 73% 77% 81% 85% 89% 93% 97%
Higher Layer Features More Invariant

\[
\|\delta^{l+1}\| = \|\sigma(z^l(v^l + \delta^l)) - \sigma(z^l(v^l))\| \approx \|\text{diag}(\sigma'(z^l(v^l(t)))) ((w^l)^T \delta^l)\| \leq \|\text{diag}(\sigma'(z^l(v^l(t)))) (w^l)^T\| \|\delta^l\|
\]

- Percentage of saturated hidden units.
- \(H<0.01\) are inactive neurons. Higher layers are more sparse.
Higher Layer Features More Invariant

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

If the norm <1 the variation shrinks one layer higher.

CD-DNN-HMM | Invariant Features | Once

| Considered Obstacles | Other Advances | Summary

Dong Yu: Keynote at IWSLT 2012
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.
Outline

- CD-DNN-HMM
- Invariant Features
- Once Considered Obstacles
- Other Advances
- 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

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

Output Layer
Hidden Layer 2
Hidden Layer 2
Hidden Layer 1
Hidden Layer 1
Input Layer

GPU3
GPU2
GPU1

Considered Obstacles
Invariant Features
Other Advances
Summary

Will cause delayed update problem since forward pass of new batch is calculated on old weight

Passing hidden activation is much more efficient than passing weight or weight gradient

Once Considered Obstacles

Will cause delayed update problem since forward pass of new batch is calculated on old weight

Passing hidden activation is much more efficient than passing weight or weight gradient
### Training (Multi-GPU): Pipeline (Chen et al. 2012)

<table>
<thead>
<tr>
<th>parallelization method</th>
<th>#GPU</th>
<th>minibatch size $T$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$K$</td>
<td>256</td>
</tr>
<tr>
<td>none (baseline)</td>
<td>1</td>
<td>68</td>
</tr>
<tr>
<td>pipeline training (0..6; 7)</td>
<td>2</td>
<td>40</td>
</tr>
<tr>
<td>vs. (0..5; 6..7)</td>
<td>2</td>
<td>36</td>
</tr>
<tr>
<td>vs. (0..2; 3..4; 5..6; 7)</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>pipeline + striped top layer</td>
<td>4</td>
<td>20</td>
</tr>
</tbody>
</table>

Multi-GPU with pipeline

- Training runtimes in minutes per 24h of data for different parallelization configurations. [[·]] denotes divergence, and [·] 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)

Lower communication cost when updating weights

Asynchronous stochastic gradient update

CD-DNN-HMM | Invariant Features | Other Advances | Summary

Asynchronous stochastic gradient update

Lower communication cost when updating weights

CD-DNN-HMM | Invariant Features | Other Advances | 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
Sequential Training

- Sequential training can achieve additional gain similar to MPE and BMMI on GMM.
- State-level minimum Bayes risk (sMBR) seems to perform better than MMI and BMMI.

**Table:** Broadcast news Dev-04f (Sainath et. al 2011)

<table>
<thead>
<tr>
<th>Training Criterion</th>
<th>1504 senones</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame-level Cross Entropy</td>
<td>18.5%</td>
</tr>
<tr>
<td>Sequence-level Criterion (sMBR)</td>
<td>17.0%</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>AM</th>
<th>Setup</th>
<th>Hub5’00-SWB</th>
<th>RT03S-FSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>SI GMM-HMM</td>
<td>BMMI+fMPE</td>
<td>18.9%</td>
<td>22.6%</td>
</tr>
<tr>
<td>SI DNN-HMM</td>
<td>7 x 2048 (frame CE)</td>
<td>16.1%</td>
<td>18.9%</td>
</tr>
<tr>
<td>SA GMM-HMM</td>
<td>BMMI+fMPE</td>
<td>15.1%</td>
<td>17.6%</td>
</tr>
<tr>
<td>SI DNN-HMM</td>
<td>7 x 2048 (sMBR)</td>
<td>13.3%</td>
<td>16.4%</td>
</tr>
</tbody>
</table>
Outline

- CD-DNN-HMM
- Invariant Features
- Once Considered Obstacles
- Other Advances
- Summary
Take Advantage of More Senones
(Li et al. 2012)

- Senone set optimized for GMM-HMM is not optimal for CD-DNN-HMM.
- **Table**: SWB WER (%). The respective optimal choices are marked in bold-face for the development set (Hub5’00-SWB).

<table>
<thead>
<tr>
<th>#sen. (J)</th>
<th>GMM-HMM (ML)</th>
<th>CD-DNN-HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gaussians</td>
<td>Hub5’00 SWB</td>
</tr>
<tr>
<td>9.0k</td>
<td>60</td>
<td>26.2</td>
</tr>
<tr>
<td>11k</td>
<td>48</td>
<td><strong>26.1</strong></td>
</tr>
<tr>
<td>15k</td>
<td>40</td>
<td>26.1</td>
</tr>
<tr>
<td>18k</td>
<td>40</td>
<td>26.1</td>
</tr>
<tr>
<td>22k</td>
<td>36</td>
<td>26.3</td>
</tr>
<tr>
<td>27k</td>
<td>28</td>
<td>26.5</td>
</tr>
<tr>
<td>32k</td>
<td>24</td>
<td>27.5</td>
</tr>
</tbody>
</table>
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.

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

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

9-13 frames of input

0.4K filters
0 or m pad

8k-Hz sampling data

CD-DNN-HMM | Invariant Features | Once Considered
Obstacles | Other Advances | Summary
## Mixed Bandwidth ASR

*(J. Li et. Al 2012)*

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

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

- **B1:** baseline 1
- **B2:** baseline 2
- **ZP:** zero padding
- **MP:** mean padding
- **UB:** upper bound

Mixed-bandwidth: recover 2/3 of (UB-B1) and ½ 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.

<table>
<thead>
<tr>
<th>Layer</th>
<th>16-kHz DNN (UB)</th>
<th>Data-mix DNN (ZP)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean (ED)</td>
<td>Variance (ED)</td>
</tr>
<tr>
<td>L1</td>
<td>13.28</td>
<td>3.90</td>
</tr>
<tr>
<td>L2</td>
<td>10.38</td>
<td>2.47</td>
</tr>
<tr>
<td>L3</td>
<td>8.04</td>
<td>1.77</td>
</tr>
<tr>
<td>L4</td>
<td>8.53</td>
<td>2.33</td>
</tr>
<tr>
<td>L5</td>
<td>9.01</td>
<td>2.96</td>
</tr>
<tr>
<td>L6</td>
<td>8.46</td>
<td>2.60</td>
</tr>
<tr>
<td>L7</td>
<td>5.27</td>
<td>1.85</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Layer</th>
<th>Mean (KL)</th>
<th>Mean (KL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top layer</td>
<td>2.03</td>
<td>0.22</td>
</tr>
</tbody>
</table>
Noise Robustness

(Look for our ICASSP 2013 paper for details)

- 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.

<table>
<thead>
<tr>
<th>Setup</th>
<th>Set A</th>
<th>Set B</th>
<th>Set C</th>
<th>Set D</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM-HMM (Baseline)</td>
<td>12.5</td>
<td>18.3</td>
<td>20.5</td>
<td>31.9</td>
<td>23.9</td>
</tr>
<tr>
<td>GMM-HMM (MPE + VAT)</td>
<td>7.2</td>
<td>12.8</td>
<td>11.5</td>
<td>19.7</td>
<td>15.3</td>
</tr>
<tr>
<td>GMM-HMM + Structured SVM</td>
<td>7.4</td>
<td>12.6</td>
<td>10.7</td>
<td>19.0</td>
<td>14.8</td>
</tr>
<tr>
<td>CD-DNN-HMM (2kx7)</td>
<td>(Look for our ICASSP 2013 paper for details)</td>
<td>13.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD-DNN-HMM (2kx7)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12.9</td>
</tr>
</tbody>
</table>
Outline

• CD-DNN-HMM
• Invariant Features
• Once Considered Obstacles
• Other Advances
• Summary
Who Can Understand Your Speech Better?

- 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
  - Rick’s demo indicates it can play important role in S2S translation
To Build a State-Of-the-Art System

- FFT
- Filter Bank
- MFCC
- Wave
- GMM
- HLDA
- VTLN
- BMMI
- fMPE

Invariants Features

Once Considered Obstacles

Other Advances

Summary
Better Accuracy and Simpler
Multilingual S2S Translation
(Look for our ICASSP 2013 paper for first step results)
Thank You
References


References

- J. Martens (2010), "Deep Learning via Hessian-free Optimization", ICML.