Deep Neural Network for Automatic Speech Recognition: from the Industry’s View

Jinyu Li
Microsoft
September 13, 2014 at Nanyang Technological University
Speech Modeling in an SR System

- Training database
- Acoustic Model Training Process
- Acoustic Model
- HMM
- Sequential Pattern Recognition (Decoding)
- Confidence Scoring
- "Hello World" (0.9) (0.8)
- Language Model
- Word Lexicon

Input Speech Flow:
- Input Speech → Feature Extraction → Acoustic Model → Confidence Scoring → "Hello World" (0.9) (0.8)
Speech Recognition and Acoustic Modeling

• SR = Finding the most probable sequence of words $W=w_1, w_2, w_3, \ldots w_n$, given the speech feature $O=o_1, o_2, o_3, \ldots o_T$

$$\text{Max}_{\{W\}} \ p(W|O)$$

$$= \text{Max}_{\{W\}} \ p(O|W)\Pr(W)/p(O)$$

$$= \text{Max}_{\{W\}} \ p(O|W)\Pr(W)$$

where

- $\Pr(W)$ : probability of $W$, computed by language model
- $p(O|W)$ : likelihood of $O$, computed by an acoustic model

• $p(O|W)$ is produced by a model $M$, $p(O|W) \rightarrow p_M(O|W)$
Challenges in Computing $P_M(O | W)$

**Model area (M):**
- Computational model: GMM/DNN
- Optimization and parameter estimation (training)
- Model recipe
- Infrastructure and engineering
- Modeling and adapting to speakers

**Feature area (O):**
- Noise-robustness
- Feature normalization algorithms
- Discriminative transformation
- Adaptation to short-term variability

**Computing $P_M(O | W)$ (run-time):**
- SVD-DNN
- Confidence/Score evaluation
- Adaptation/Normalization
- Quantization
Acoustic Modeling of a Word

- Hidden Markov model (HMM)
- State emission distribution is modeled by DNN or GMM

Tri-phone representation of “it”
DNN for Automatic Speech Recognition

- **DNN**
  - Feed-forward artificial neural network
  - More than one layer of hidden units between input and output
  - Apply a nonlinear/linear function in each layer

- **DNN for automatic speech recognition (ASR)**
  - Replace the Gaussian mixture model (GMM) in the traditional system with a DNN to evaluate state likelihood
Phoneme State Likelihood Modeling

Phoneme State Likelihood Modeling

Phoneme State Likelihood Modeling

Phoneme State Likelihood Modeling

DNN Fundamental Challenges to Industry

1. How to reduce the runtime without accuracy loss?
2. How to do speaker adaptation with low footprints?
3. How to be robust to noise?
4. How to reduce accuracy gap between large and small DNN?
5. How to deal with large variety of data?
6. How to enable languages with limited training data?
Reduce DNN Runtime without Accuracy Loss
The runtime cost of DNN is much larger than that of GMM, which has been fully optimized in product deployment. We need to reduce the runtime cost of DNN in order to ship it.
Solution

- The runtime cost of DNN is much larger than that of GMM, which has been fully optimized in product deployment. We need to reduce the runtime cost of DNN in order to ship it.
- We propose a new DNN structure by taking advantage of the low-rank property of DNN model to compress it.
Singular Value Decomposition (SVD)

\[ A_{m \times n} = U_{m \times n} \Sigma_{n \times n} V_{n \times n}^T = \begin{bmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mn} \end{bmatrix} \begin{bmatrix} \epsilon_{11} & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & \epsilon_{kk} & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & \epsilon_{nn} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{n1} & \cdots & v_{nn} \end{bmatrix} \]
SVD Approximation

- Number of parameters: $mn \rightarrow mk+nk$.
- Runtime cost: $O(mn) \rightarrow O(mk+nk)$.
- E.g., $m=2048$, $n=2048$, $k=192$. 80% runtime cost reduction.
SVD-Based Model Restructuring
SVD-Based Model Restructuring
SVD-Based Model Restructuring
Proposed Method

- Train standard DNN model with regular methods: pre-training + cross entropy fine-tuning
- Use SVD to decompose each weight matrix in standard DNN into two smaller matrices
- Apply new matrices back
- Fine-tune the new DNN model if needed
## A Product Setup

<table>
<thead>
<tr>
<th>Acoustic model</th>
<th>WER</th>
<th>Number of parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original DNN model</td>
<td>25.6%</td>
<td>29M</td>
</tr>
<tr>
<td>SVD (512) to hidden layer</td>
<td>25.7%</td>
<td>21M</td>
</tr>
<tr>
<td>All hidden and output layer (192)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Before fine-tune</td>
<td>36.7%</td>
<td>5.6M</td>
</tr>
<tr>
<td>After fine-tune</td>
<td>25.5%</td>
<td></td>
</tr>
</tbody>
</table>

Around 80% runtime cost reduction!
Adapting DNN to Speakers with Low Footprints

[Xue 14]
Motivation

- Speaker personalization with a DNN model creates a storage size issue: It is not practical to store an entire DNN model for each individual speaker during deployment.
Solution

- Speaker personalization with a DNN model creates a storage size issue: It is not practical to store an entire DNN model for each individual speaker during deployment.

- We propose low-footprint DNN personalization method based on SVD structure.
SVD Personalization

- SVD Restructure: $A_{m\times n} \approx U_{m\times k} W_{k\times n}$
- SVD Personalization: $A_{m\times n} \approx U_{m\times k} S_{k\times k} W_{k\times n}$. Initiate $S_{k\times k}$ as $I_{k\times k}$, and then only adapt/store the speaker-dependent $S_{k\times k}$. 
SVD Personalization Structure
SVD Personalization Structure
Adapt with 100 Utterances

<table>
<thead>
<tr>
<th></th>
<th>Full-rank SI model</th>
<th>SVD model</th>
<th>Standard adaptation</th>
<th>SVD adaptation</th>
</tr>
</thead>
<tbody>
<tr>
<td>WER</td>
<td>25.21%</td>
<td>25.12%</td>
<td>20.51%</td>
<td>19.95%</td>
</tr>
<tr>
<td>Number of parameters (M)</td>
<td>30</td>
<td>7.4</td>
<td>7.4</td>
<td>0.26</td>
</tr>
</tbody>
</table>
Noise Robustness

[Li14, Zhao 14, Zhao 14b]
DNN Is More Robust to Distortion – Multi-condition-trained DNN on Training Utterances
DNN Is More Robust to Distortion – Multi-condition-trained DNN on Training Utterances
DNN Is More Robust to Distortion – Multi-condition-trained DNN on Training Utterances
DNN Is More Robust to Distortion – Multi-condition-trained DNN on Training Utterances
Noise-Robustness Is Still Most Challenging – Clean-trained DNN on Test Utterances
Noise-Robustness Is Still Most Challenging – Clean-trained DNN on Test Utterances
Noise-Robustness Is Still Most Challenging – Clean-trained DNN on Test Utterances
Noise-Robustness Is Still Most Challenging – Multi-condition-trained DNN on Test Utterances
Noise-Robustness Is Still Most Challenging – Multi-condition-trained DNN on Test Utterances
Noise-Robustness Is Still Most Challenging – Multi-condition-trained DNN on Test Utterances
Some Observations

- DNN works very well on utterances and environments observed.
- For the unseen test case, DNN cannot generalize very well. Therefore, noise-robustness technologies are still important.
- For more technologies on noise-robustness, refer to our recent overview paper [Li14] for more studies.
Variable Component DNN

- DNN components:
  - Weight matrices, outputs of a hidden layer.

- For any of the DNN components
  - Training: Model it as a set of polynomial functions of a context variable, e.g. SNR, duration, speaking rate.
    \[ C_l = \sum_{j=0}^{J} C_j^l v_j \quad 0 < l \leq L \quad (J \text{ is the order of polynomials}) \]
  - Recognition: compute the component on-the-fly based on the variable and the associated polynomial functions.

- Developed VP-DNN, VO-DNN.
VPDNN
VPDNN Improves Robustness on Noisy Environment Un-seen in the Training

- The training data has SNR > 10db.

<table>
<thead>
<tr>
<th></th>
<th>5dB-10dB</th>
<th>&gt; 10dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>WER(%)</td>
<td>standard DNN</td>
<td>VPDNN</td>
</tr>
<tr>
<td>Average</td>
<td>13.85</td>
<td>12.68</td>
</tr>
<tr>
<td>Relative WERR(%)</td>
<td>8.47%</td>
<td></td>
</tr>
</tbody>
</table>
Reduce Accuracy Gap between Large and Small DNN

[Li14c]
To Deploy DNN on Server

- Low rank matrices are used to reduce the number of DNN parameters and CPU cost.
- Quantization for SSE evaluation is used for single instruction multiple data processing.
- Frame skipping or prediction is used to remove the evaluation of some frames.
To Deploy DNN on Device

- The industry has strong interests to have DNN systems on devices due to the increasingly popular mobile scenarios.
- Even with the technologies mentioned above, the large computational cost is still very challenging due to the limited processing power of devices.
- A common way to fit CD-DNN-HMM on devices is to reduce the DNN model size by
  - reducing the number of nodes in hidden layers
  - reducing the number of senone targets in the output layer
- However, these methods significant increase word error rate.
- **In this talk, we explore a better way to reduce the DNN model size with less accuracy loss than the standard training method.**
Standard DNN Training Process

- Generate a set of senones as the DNN training target: splits the decision tree by maximizing the increase of likelihood evaluated on single Gaussians
- Get transcribed training data
- Train DNN with cross entropy or sequence training criterion
Significant Accuracy Loss when DNN Size Is Significantly Reduced

- Better accuracy is obtained if we use the output of large-size DNN for acoustic likelihood evaluation.
- The output of small-size DNN is away from that of large-size DNN, resulting in worse recognition accuracy.
- The problem is solved if the small-size DNN can generate similar output as the large-size DNN.
Can We Make the Small-size DNN Generate Similar Output to the Large-size DNN?

- No -- if we only have transcribed data.
- Yes -- in industry, we have almost unlimited un-transcribed data and only a small portion is transcribed
Small-Size DNN Training with Output Distribution Learning

- Use the standard DNN training method to train a large-size teacher DNN using transcribed data
- Random initialize the small-size student DNN
- Minimize the KL divergence between the output distribution of the student DNN and teacher DNN with large amount of un-transcribed data
Minimize the KL Divergence between the Output Distribution of DNNs

\[
\sum_t \sum_{i=1}^N P_L(s_i|x_t) \log \left( \frac{P_L(s_i|x_t)}{P_S(s_i|x_t)} \right) - \sum_t \sum_{i=1}^N P_L(s_i|x_t) \log P_S(s_i|x_t)
\]

- \(s_i\): \(i\)-th senone
- \(x_t\): the observation at time \(t\)
- \(P_L(s_i|x_t), P_S(s_i|x_t)\): posterior output distribution of teacher and student DNN, respectively

- A general form of the standard DNN training criterion where the target is a one-hot vector.
- Here the target is generated by the output of teacher DNN
Experiment Setup

- 375 hours of transcribed US-English data
- Large-size DNN: 5*2048
- Small-size DNN: 5*512
- 6k senones
## EN-US Windows Phone Task

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>19.90</td>
</tr>
</tbody>
</table>
EN-US Windows Phone Task

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>19.90</td>
</tr>
</tbody>
</table>
EN-US Windows Phone Task

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>19.90</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>19.55</td>
</tr>
</tbody>
</table>
**EN-US Windows Phone Task**

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>19.90</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>19.55</td>
</tr>
<tr>
<td>5 * 512</td>
<td>750 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>19.28</td>
</tr>
</tbody>
</table>
EN-US Windows Phone Task

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>19.90</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>19.55</td>
</tr>
<tr>
<td>5 * 512</td>
<td>750 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>19.28</td>
</tr>
<tr>
<td>5 * 512</td>
<td>1500 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>18.89</td>
</tr>
</tbody>
</table>
## EN-US Windows Phone Task

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard cross entropy</td>
<td>19.90</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>19.55</td>
</tr>
<tr>
<td>5 * 512</td>
<td>750 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>19.28</td>
</tr>
<tr>
<td>5 * 512</td>
<td>Decode 750 hours un-transcribed data to generate transcription</td>
<td>Standard cross entropy</td>
<td>20.48</td>
</tr>
</tbody>
</table>
### Can We Use German Data to Learn EN-US DNN?

Use it as the teacher for output distribution learning.

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>TrainingCriterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours EN-US transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>750 hours un-transcribed EN-US data</td>
<td>Output distribution learning</td>
<td>19.28</td>
</tr>
<tr>
<td>5 * 512</td>
<td>600 hours un-transcribed German data</td>
<td>Output distribution learning</td>
<td>?</td>
</tr>
</tbody>
</table>
Can We Use German Data to Learn EN-US DNN?

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours EN-US transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>750 hours un-transcribed EN-US data</td>
<td>Output distribution learning</td>
<td>19.28</td>
</tr>
<tr>
<td>5 * 512</td>
<td>600 hours un-transcribed German data</td>
<td>Output distribution learning</td>
<td>?</td>
</tr>
</tbody>
</table>

Please guess a WER
90? 70? 50? 30? 10?
Can We Use German Data to Learn EN-US DNN?

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours EN-US transcribed data</td>
<td>Standard cross entropy</td>
<td>16.32</td>
</tr>
<tr>
<td>5 * 512</td>
<td>750 hours un-transcribed EN-US data</td>
<td>Output distribution learning</td>
<td>19.28</td>
</tr>
<tr>
<td>5 * 512</td>
<td>600 hours un-transcribed German data</td>
<td>Output distribution learning</td>
<td>21.71</td>
</tr>
</tbody>
</table>
Better Teacher

- If the teacher DNN is improved by some other techniques, could the improvement be transferred to a better student DNN?
Better Teacher

- If the teacher DNN is improved by some other techniques, could the improvement be transferred to a better student DNN?

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard sequence training</td>
<td>13.93</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard sequence training</td>
<td>17.16</td>
</tr>
</tbody>
</table>
If the teacher DNN is improved by some other techniques, could the improvement be transferred to a better student DNN?

Use it as the teacher for output distribution learning

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Training Criterion</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 * 2048</td>
<td>375 hours transcribed data</td>
<td>Standard sequence training</td>
<td>13.93</td>
</tr>
<tr>
<td>5 * 512</td>
<td>375 hours transcribed data</td>
<td>Standard sequence training</td>
<td>17.16</td>
</tr>
<tr>
<td>5 * 512</td>
<td>750 hours un-transcribed data</td>
<td>Output distribution learning</td>
<td>16.66</td>
</tr>
</tbody>
</table>
Real Application Setup

- 2 Million parameter for small-size DNN, compared to 30 Million parameters for teacher DNN

Teacher DNN trained with standard sequence training

Student DNN trained with output distribution learning in this talk

Small-size DNN trained with standard sequence training
Dealing with Large Variety of Data

[Li 12, 14b]
Factorization of Speech Signals

\[ R(x) = R(y) + \sum_{n=1}^{N} Q_n f_n, \]
Joint Factor Analysis (JFA)-Style Adaptation

• JFA: $M = m + Aa + Bb + Cc,$

$$R(x) \approx R(y) + Dn + Eh + Fs$$
Vector Tayler Series (VTS)-Style Adaptation

\[ x = y + \log(1 - \exp(n - y)) \]
\[ \approx y + \log(1 - \exp(n_0 - y_0)) + A(y - y_0) + B(n - n_0) \]
\[ R(x) \approx R(y) + \frac{\partial R}{\partial y} (Ay + Bn + \text{const.}) \]
If we make a rather coarse assumption that \( \partial R/\partial y \) is constant
\[ R(x) \approx R(y) + Cy + Dn + \text{const} \]
Fast Adaptation with Factorization

**Test set B – same microphone**

**Test set D – microphone mismatch**
Factorization of Speech Signals, Another Solution
DNN SR for 8-kHz and 16-kHz Data
Performance on Wideband and Narrowband Test Sets

<table>
<thead>
<tr>
<th>Training Data</th>
<th>WER (16-kHz)</th>
<th>WER (8-kHz)</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 + 16-kHz VS-2 (UB)</td>
<td>27.47</td>
<td>53.51</td>
</tr>
</tbody>
</table>
Distance for the Output Vectors between 8-kHz and 16-kHz Input Features
Enable Languages with Limited Training Data
Shared Hidden Layer Multi-lingual DNN

Input Layer:
A window of acoustic feature frames

Many Hidden Layers

Language 1 senones
Language 2 senones
Language 3 senones
Language 4 senones

Shared Feature Transformation

Lang 1  Lang 2  Lang 3  Lang 4  Training or Testing Samples
Source Languages in Multilingual DNN Benefit Each Other

<table>
<thead>
<tr>
<th></th>
<th>FRA</th>
<th>DEU</th>
<th>ESP</th>
<th>ITA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Set Size (Words)</td>
<td>40K</td>
<td>37K</td>
<td>18K</td>
<td>31K</td>
</tr>
<tr>
<td>Monolingual DNN</td>
<td>28.1</td>
<td>24.0</td>
<td>30.6</td>
<td>24.3</td>
</tr>
<tr>
<td>SHL-DNN</td>
<td>27.1</td>
<td>22.7</td>
<td>29.4</td>
<td>23.5</td>
</tr>
<tr>
<td>Relative WER Reduction</td>
<td>3.6</td>
<td>5.4</td>
<td>3.9</td>
<td>3.3</td>
</tr>
</tbody>
</table>

source languages: FRA: 138 hours, DEU: 195 hours, ESP: 63 hours, and ITA: 93 hours of speech.
Transferring from Western Languages to Mandarin Chinese Is Effective

<table>
<thead>
<tr>
<th>CHN CER (%)</th>
<th>3 hrs</th>
<th>9hrs</th>
<th>36hrs</th>
<th>139hrs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline DNN (no transfer)</td>
<td>45.1</td>
<td>40.3</td>
<td>31.9</td>
<td>29.0</td>
</tr>
<tr>
<td>SHL-MDNN Model Transfer</td>
<td>35.6</td>
<td>33.9</td>
<td>28.4</td>
<td>26.6</td>
</tr>
<tr>
<td>Relative CER Reduction</td>
<td>21.1</td>
<td>15.9</td>
<td>10.4</td>
<td>8.3</td>
</tr>
</tbody>
</table>

source languages: FRA: 138 hours, DEU: 195 hours, ESP: 63 hours, and ITA: 93 hours of speech.
[Huang 13] Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong, cross-language knowledge transfer using multilingual deep neural network with shared hidden layers, in ICASSP, 2013

[Li12] Jinyu Li, Dong Yu, Jui-Ting Huang, and Yifan Gong, improving wideband speech recognition using mixed-bandwidth training data in CD-DNN-HMM, in IEEE Workshop on Spoken Language Technology, 2012


[Li14b] Jinyu Li, Jui-Ting Huang, and Yifan Gong, Factorized adaptation for deep neural network, in ICASSP, 2014

[Li14c] Jinyu Li, Rui Zhao, Jui-Ting Huang and Yifan Gong, Learning Small-Size DNN with Output-Distribution-Based Criteria, in Interspeech, 2014.


[Xue 14] Jian Xue, Jinyu Li, Dong Yu, Mike Seltzer, and Yifan Gong, Singular Value Decomposition Based Low-footprint Speaker Adaptation and Personalization for Deep Neural Network, in ICASSP, 2014

[Zhao14] Rui Zhao, Jinyu Li and Yifan Gong, Variable-Component Deep Neural Network for Robust Speech Recognition, in Interspeech, 2014.