Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition
Paul Hensch
Large-Vocabulary **Speech Recognition**

**Complications**

- **Speaker**
  - Accents
  - Dialect
  - Style
  - Emotion
  - Coarticulation
  - Reduction
  - Pronunciation
  - Hesitation
  - ...

- **Environment**
  - Noise
  - Side talk
  - Reverberation
  - ...

- **Device**
  - Head phone
  - Land phone
  - Speaker phone
  - Cell phone
  - ...

21.01.2014 | Seminar aus maschinellem Lernen | 2
Large-Vocabulary **Speech Recognition**

- **Hidden Markov Models**
  - Find Phones, which matches input

- **Neural networks**
  - Look on frequency changes over time to recognize

- **Hybrid**
  - CD-GMM-HMM
  - CD-DNN-HMM
CD-DNN-HMM

- **Similar classes of sounds, or phones**
  - Diphone
  - Triphone
  - Quinphone

- **Senons are used as the DNN output unit**
Deep Belief Network
(not dynamic Bayes net)

2 Stages:
- Pre-training
- Fine-tuning
Pre-training advantages:

- Often also achieve lower training error
- Sort of data-dependent regularization
Pre-training

learning the connection weights in a DBN that is equivalent to training each adjacent pair of layers as an restricted Boltzmann machine (RBM).
Last steps of training:

With pre-training complete, add a randomly initialized softmax output layer and use backpropagation to fine-tune all the weights.
Hidden Markov Models is the dominant technique for LVSR
CD-DNN-HMM
More advantages:

- Implement a CD-DNN-HMM system with only minimal modifications to an existing CD-GMM-HMM system
- Any improvements in modeling units that are incorporated into the CD-GMM-HMM baseline system, such as cross-word triphone models, will be accessible to the DNN
EXPERIMENTAL RESULTS

Business search dataset collected from the Bing mobile voice search application

- Collected under real usage scenarios in 2008
- Sampled at 8 kHz
- Encoded with the GSM codec
- contains all kinds of variations: noise, music, side-speech, accents, sloppy pronunciation …
EXPERIMENTAL RESULTS

INFORMATION ON THE BUSINESS SEARCH DATASET

<table>
<thead>
<tr>
<th>Set</th>
<th>Hours</th>
<th>Number of Utterances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Set</td>
<td>24</td>
<td>32,057</td>
</tr>
<tr>
<td>Development Set</td>
<td>6.5</td>
<td>8,777</td>
</tr>
<tr>
<td>Test Set</td>
<td>9.5</td>
<td>12,758</td>
</tr>
</tbody>
</table>

- Dataset contains 65 K word unigrams, 3.2 million word bi-grams, and 1.5 million word tri-grams.
- Sentence length is 2.1 tokens
EXPERIMENTAL RESULTS

Compare sentences (sentence accuracy) instead of word accuracy.

G. Zweig and P. Nguyen, “A segmental CRF approach to large vocabulary continuous speech recognition”

Difficulties:

• “Mc-Donalds” “McDonalds”
• “Walmart” “Wal-mart”
• “7-eleven” “7 eleven” “seven-eleven.”

Maximum of 94% accuracy.
EXPERIMENTAL RESULTS

Baseline Systems

- trained clustered cross-word triphone GMM-HMM
- The performance of the best CD-GMM-HMM configuration is summarized in Table

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Dev Accuracy</th>
<th>Test Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML</td>
<td>62.9%</td>
<td>60.4%</td>
</tr>
<tr>
<td>MMI</td>
<td>65.1%</td>
<td>62.8%</td>
</tr>
<tr>
<td>MPE</td>
<td>65.5%</td>
<td>63.8%</td>
</tr>
</tbody>
</table>
EXPERIMENTAL RESULTS

CD-DNN-HMM Results and Analysis

Comparison of Context-Independent Monophone State Labels and Context-Dependent Triphone Senone Labels

<table>
<thead>
<tr>
<th># Hidden Layers</th>
<th># Hidden Units</th>
<th>Label Type</th>
<th>Dev Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2K</td>
<td>Monophone States</td>
<td>59.3%</td>
</tr>
<tr>
<td>1</td>
<td>2K</td>
<td>Triphone Senones</td>
<td>68.1%</td>
</tr>
<tr>
<td>3</td>
<td>2K</td>
<td>Monophone States</td>
<td>64.2%</td>
</tr>
<tr>
<td>3</td>
<td>2K</td>
<td>Triphone Senones</td>
<td>69.6%</td>
</tr>
</tbody>
</table>
### Experimental Results

**CD-DNN-HMM Results and Analysis**

#### Context-Dependent Models With and Without Pre-Training

<table>
<thead>
<tr>
<th>Model Type</th>
<th># Hidden Layers</th>
<th># Hidden Units</th>
<th>Dev Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>without pre-training</td>
<td>1</td>
<td>2K</td>
<td>68.0%</td>
</tr>
<tr>
<td>without pre-training</td>
<td>2</td>
<td>2K</td>
<td>68.2%</td>
</tr>
<tr>
<td>with pre-training</td>
<td>1</td>
<td>2K</td>
<td>68.1%</td>
</tr>
<tr>
<td>with pre-training</td>
<td>2</td>
<td>2K</td>
<td>69.5%</td>
</tr>
</tbody>
</table>
EXPERIMENTAL RESULTS

CD-DNN-HMM Results and Analysis

Relationship between the recognition accuracy and the number of layers. Context-dependent models with 2 K hidden units
EXPERIMENTAL RESULTS

Training and Decoding Time

• Trainer written in Python
• Carried out on Dell Precision T3500 workstation
  • Quad core computer
  • CPU clock speed of 2.66 GHz, 8 MB of L3 CPU cache
  • 12 GB of 1066 MHz DDR3 SDRAM.
  • NVIDIA Tesla C1060 (GPGPU), which contains 4 GB of GDDR3 RAM and 240 processing cores
## EXPERIMENTAL RESULTS

### Training and Decoding Time

**Summary of Training Time Using 24 Hours of Training Data and 2 K Hidden Units Per Layer**

<table>
<thead>
<tr>
<th>Type</th>
<th># of Layers</th>
<th>Time Per Epoch</th>
<th># of Epochs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-train</td>
<td>1</td>
<td>0.2 h</td>
<td>50</td>
</tr>
<tr>
<td>Pre-train</td>
<td>2</td>
<td>0.5 h</td>
<td>20</td>
</tr>
<tr>
<td>Pre-train</td>
<td>3</td>
<td>0.6 h</td>
<td>20</td>
</tr>
<tr>
<td>Pre-train</td>
<td>4</td>
<td>0.7 h</td>
<td>20</td>
</tr>
<tr>
<td>Pre-train</td>
<td>5</td>
<td>0.8 h</td>
<td>20</td>
</tr>
<tr>
<td>Fine-tune</td>
<td>4</td>
<td>1.2 h</td>
<td>12</td>
</tr>
<tr>
<td>Fine-tune</td>
<td>5</td>
<td>1.4 h</td>
<td>12</td>
</tr>
</tbody>
</table>
EXPERIMENTAL RESULTS

Training and Decoding Time

<table>
<thead>
<tr>
<th>Processing Unit</th>
<th># of Layers</th>
<th>DNN Time Per Frame</th>
<th>Search Time Per Frame</th>
<th>Real-time Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>4</td>
<td>4.3 ms</td>
<td>1.5 ms</td>
<td>0.58</td>
</tr>
<tr>
<td>GPU</td>
<td>4</td>
<td>0.16 ms</td>
<td>1.5 ms</td>
<td>0.17</td>
</tr>
<tr>
<td>CPU</td>
<td>5</td>
<td>5.2 ms</td>
<td>1.5 ms</td>
<td>0.67</td>
</tr>
<tr>
<td>GPU</td>
<td>5</td>
<td>0.20 ms</td>
<td>1.5 ms</td>
<td>0.17</td>
</tr>
</tbody>
</table>

• five-layer CD-DNN-HMM, pre-training takes
  \[0.2 \times 50 + 0.5 \times 20 + 0.6 \times 20 + 0.7 \times 20 + 0.8 \times 20 = 62\text{ hours}\]

• Fine-tuning takes
  \[1.4 \times 12 = 16.8\text{ hours}\]
EXPERIMENTAL RESULTS

Training and Decoding Time

Observations:
The bottleneck in the training process is the mini-batch stochastic gradient descend (SGD) algorithm used to train the DNNs.

It is extrapolated that using similar technics described in this paper, it should be possible to train an effective CD-DNN-HMM system that exploits 2000 hours of training data in about 50 days.
CONCLUSION AND FUTURE WORK

- CD-DNN-HMM is more expensive than GMM
- CD-DNN-HMM performs better than GMM
- Finding new ways to parallelize training
- Finding highly effective speaker and environment adaptation algorithms for DNN-HMMs
- The training in this study used the embedded Viterbi algorithm, which is not optimal (MFCC)
Thank your for your attention

Questions?..