Deep Neural Networks and their Applicability to Speaker De-Identification

Dr. László Tóth

MTA-SZTE Research Group on Artificial Intelligence
Szeged, Hungary
Neural Nets in Speech Recognition

- ANN’s are present in speech recognition since the late eighties
- The so-called HMM/ANN hybrid model achieved the best results
  - The ANN performs the local probability estimation
  - Search and combination is done by the same old HMM
- Somewhat better results than standard HMM (mainly on smaller tasks), but no breakthrough
The Quick Success of Deep Neural Nets

2006: DNNs first published (Science, image test data)
2009: First application to speech recognition
  - new record on TIMIT
2012:  - Two dedicated sections at Interspeech
  - Google and Microsoft claims to achieve 10-30% error rate reduction in their products thanks to DNNs
2014: 4 DNN sections at ICASSP
  - About 50 papers containing ”deep neural” in their title
Conventional vs. „deep” ANN

Conventional ANN: one hidden layer
- training with backpropagation (gradient descent)

Why just one hidden layer?
- it already allows arbitrary precision (with the no. of neurons going to infinity…)
- it is already too slow
Conventional vs. „deep” ANN

Deep ANN: more (2-9) hidden layers
- if the number of hidden neurons is fixed, it is better to use many small hidden layers than just one huge

#1. Problem: very slow
- solution: use GPUs (20-40 times speedup in matrix multiplication!)

#2. Problem: backprop is not efficient with many layers
- „explaining away” effect
- „vanishing gradient” effect
Solutions for training difficulties

1. „Pre-training” with new algorithms
   - standard: CD-pretraining with DBNs (Hinton et al., 2006)
   - many newer methods proposed since then
2. Newer types of neurons and network architectures
   - rectified linear neurons
   - maxout neurons
   - convolutional neural nets
The Restricted Boltzmann Machine (RBM)

Similar to a 2-layer ANN

- but takes binary values rather than continuous

Training: contrastive divergence (CD)

- unsupervised
- aims to reconstruct its input from the hidden units
- CD is similar to Maximum Likelihood
- training is iterative, just as backpropagation
Deep Belief Network

Deep Belief Network: RBMs on top of each other

- training: CD algorithm, adding layer by layer
Conversion Into a Deep ANN

How to turn the DBN into a DNN?
- Replacing RBMs by standard neurons (using the same weights)
- Adding an uppermost softmax layer
- backpropagation training using the class target labels

Problems
- The CD algorithm is slow and complicated
Results on TIMIT

For 4 hidden layers it agrees with the literature, but worse for 5 layers.
Results on Hungarian Broadcast News Dataset

![Graph showing phone error rate vs. number of hidden layers for pre-training and no pre-training.]
Discriminative Pretraining

The ANN is grown layer by layer

- training: backpropagation (no new algorithm required!)
- Train just a couple of iterations after the addition of each layer
Rectifier neural networks (Glorot, 2011)

It modifies the neurons, and not the training method
  - *The sigmoid activation function is replaced by max(0,x)*
  - *Training with backpropagation*
  - *Keeping the weights within limits by normalization*
Rectifier neural nets (2)

Experience: rectifier nets can achieve the same accuracy using only backprop, than sigmoid nets with CD-pretraining

- *we were among the first ones who applied rectifier nets for ASR*

Phone recognition with deep sparse rectifier neural networks
L Tóth - Acoustics, Speech and Signal Processing (ICASSP), …, 2013 - ieeexplore.ieee.org
Idézetek száma: 12  Kapcsolódó cikkek  Mind a(z) 3 változat  Idézés  Mentés  Tovább

On rectified linear units for speech processing
MD Zeiler, M Ranzato, R Monga, M Mao… - …, Speech and Signal …, 2013 - ieeexplore.ieee.org
Idézetek száma: 44  Kapcsolódó cikkek  Mind a(z) 16 változat  Idézés  Mentés  Tovább

Improving deep neural networks for LVCSR using rectified linear units and dropout
Idézetek száma: 72  Kapcsolódó cikkek  Mind a(z) 9 változat  Idézés  Mentés  Tovább
Results on TIMIT
Results on the Broadcast News Dataset

Comment: HMM (with HTK): 19.9% (16% WER reduction)
A Comparison of Training Times

<table>
<thead>
<tr>
<th>Training method</th>
<th>Pre-training time</th>
<th>Fine-tuning training time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBN pre-training</td>
<td>48 hours</td>
<td>14 hours</td>
</tr>
<tr>
<td>Discr. pre-training</td>
<td>9 hours</td>
<td>11 hours</td>
</tr>
<tr>
<td>Rectifier network</td>
<td>0 hours</td>
<td>14.5 hours</td>
</tr>
</tbody>
</table>

Table 1. The training times required by the various methods for 5-layer networks.
Convolutional Neural Nets

• An old method in image processing, but quite new in ASR (only 10-20 papers so far)

• Convolutional neuron $\leftrightarrow$ conventional neuron:
  • The input of the neurons is localized
  • The same neuron is evaluated at several slightly shifted positions
  • The results are „pooled”
    • Most frequent: max pooling
Convolution in Speech Recognition

- Input: mel-frequency band energies
- The freq. axis is divided into wider bands
- Convolution is performed along frequency
- Higher layers: fully connected

- The effect of convolution: helps handle speaker and speaking style variations
Results (TIMIT)

<table>
<thead>
<tr>
<th>Number of filters</th>
<th>Width of filters</th>
<th>Number of units per filter</th>
<th>Error on devel. set</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>12</td>
<td>768</td>
<td>16.6%</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>638</td>
<td>16.6%</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>554</td>
<td>16.9%</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>485</td>
<td>16.6%</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>433</td>
<td>16.5%</td>
</tr>
</tbody>
</table>

Table 2. Phone error rates obtained by applying convolution along frequency, using various sets of convolutional filters.

Fig. 2. The influence of pooling size on the phone error rate.

9% error reduction compared to the fully connected deep network.
Convolution along the time axis

- Source: Vesely (2011)
- The time axis is divided into blocks
- These blocks are processed by the same sub-network
- This is a refined version of the „stacked” or „hierarchical” processing method
## Results (TIMIT)

<table>
<thead>
<tr>
<th>Sizes of hidden layers</th>
<th>Context</th>
<th>Error on dev. set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>local</td>
<td>conv.</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>9 frames</td>
<td>0, \pm 3, \pm 6</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>9 frames</td>
<td>0, \pm 4, \pm 8</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>9 frames</td>
<td>0, \pm 5, \pm 10</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>13 frames</td>
<td>0, \pm 3, \pm 6</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>13 frames</td>
<td>0, \pm 4, \pm 8</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>13 frames</td>
<td>0, \pm 5, \pm 10</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>17 frames</td>
<td>0, \pm 3, \pm 6</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>17 frames</td>
<td>0, \pm 4, \pm 8</td>
</tr>
<tr>
<td>2000-400-2\cdot2000</td>
<td>17 frames</td>
<td>0, \pm 5, \pm 10</td>
</tr>
<tr>
<td>2000-1000-2\cdot2000</td>
<td>9 frames</td>
<td>0, \pm 5, \pm 10</td>
</tr>
</tbody>
</table>

9% error reduction compared to the fully connected deep network
Convolution along both axes

<table>
<thead>
<tr>
<th>Network topology</th>
<th>devel. set</th>
<th>core test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>convolution along both axes</td>
<td>14.2%</td>
<td>17.6%</td>
</tr>
</tbody>
</table>

- This is one of the best results on TIMIT
- Graves et al.: 17.7% using RNNs
- Peddinti et al.: 17.4% using CNNs with scattering features
Goal: to decrease overfitting

Dropout (Hinton et al., 2012): during training, $x$ percent of the neural outputs are turned to zero („dropped out”)

- In the original paper $x=50\%$ for others $x=10-20\%$
- Indeed increases generalization (better results on the test set), but the training time is also increased (about 3-5 times)
Results with convolution plus dropout

<table>
<thead>
<tr>
<th>Network topology</th>
<th>devel. set</th>
<th>core test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>convolution along both axes</td>
<td>14.2%</td>
<td>17.6%</td>
</tr>
<tr>
<td>the above plus dropout</td>
<td>13.9%</td>
<td>16.7%</td>
</tr>
</tbody>
</table>

- Currently the best result on TIMIT
Results on an LVCSR Task

- In cooperation with the Technical University of Budapest
Summary

- DNN/HMM hybrids offer 20-30% error rate reduction over standard HMMs in LVCSR
  - We obtained similar improvements
- CNNs offer 10% improvement even over DNNs
  - We have our own code, which holds the record of the lowest phone error rate on TIMIT
- We are looking for further possible applications and partnerships
DNNs for Speaker De-Identification

• Approach #1: ASR → phone sequence → TTS

• Advantages:
  • Discards all acoustic features of the speaker
  • No need to modify ASR and TTS code

• Disadvantage: High phone error rate of ASR
  • However, currently we have the best phone recognizer…
  • Dobrišek et al. found that the synthetized speech is quite intelligible, as the ASR substitutes phonetically similar phones (Intelligibility Assessment of De-Identified Speech Obtained Using Phone Recognition and Speech Synthesis Systems, Proc. TSD 2014)
Using Other Abstract Representation

• Idea #2: ASR \(\rightarrow\) abstract features \(\rightarrow\) TTS
  • The abstraction should not go to the level of phones, but can be any other speaker independent features
• ASR \(\rightarrow\) articulatory features \(\rightarrow\) TTS
  • Sondhi et al.: On the use of neural networks in articulatory speech synthesis. JASA 1993
• ASR \(\rightarrow\) synthesis parameters \(\rightarrow\) TTS
  • See the presentation of Gerard Chollet…
Using DBNs in Generative Mode

• Idea #3: DBNs can be used to generate signals
  • DBNs are trained to reconstruct their input
  • So they can be used in “reconstruction mode” for any input
  • If trained on many speakers, they would theoretically return some idealized, abstracted “average voice”
    • I’ve found such examples for images, but not for speech
    • Some features other than MFCC should be used. Luckily, DNNs work quite well even with raw acoustic signal
Using DBNs for Voice Conversion

• Idea #4: Map the source voice directly to target voice
  • Conventionally, the mapping transformation is learned by GMM-based methods
  • Recently, much results are obtained with DNNs