Deep Neural Network Acoustic Models

Steve Renals

Automatic Speech Recognition – ASR Lecture 12
25 February 2016
Recap
Hybrid NN/HMM

"Don’t Ask"

DON’T

d

oh

n

t

ASK

ah

s

k

Utterance

Word

Subword (phone)

Acoustic model (HMM)

Speech Acoustics

3x39 = 117 phone states

~1000 hidden units

1 hidden layer

9x39 MFCC inputs

x(t-4) x(t-3) x(t) x(t+3) x(t+4)
Advantages of NN:
- Can easily model \textit{correlated features}
  - Correlated feature vector components (eg spectral features)
  - Input context – multiple frames of data at input
- \textbf{More flexible} than GMMs – not made of (nearly) local components; GMMs inefficient for non-linear class boundaries
- NNs can \textbf{model multiple events} in the input simultaneously – different sets of hidden units modelling each event; GMMs assume each frame generated by a single mixture component.
- NNs can \textbf{learn richer representations} and learn ‘higher-level’ features (tandem, posteriorgrams, bottleneck features)
Advantages of NN:
- Can easily model **correlated features**
  - Correlated feature vector components (e.g., spectral features)
  - Input context – multiple frames of data at input
- **More flexible** than GMMs – not made of (nearly) local components; GMMs inefficient for non-linear class boundaries
- NNs can **model multiple events** in the input simultaneously – different sets of hidden units modelling each event; GMMs assume each frame generated by a single mixture component.
- NNs can **learn richer representations** and learn ‘higher-level’ features (tandem, posteriorgrams, bottleneck features)

Disadvantages of NN:
- Until ∼ 2012:
  - Context-independent (monophone) models, weak speaker adaptation algorithms
  - NN systems less complex than GMMs (fewer parameters):
    - RNN – < 100k parameters, MLP – ∼ 1M parameters
- Computationally expensive - more difficult to parallelise training than GMM systems
Deep Neural Network
Acoustic Models
Deep neural networks (DNNs) — Hybrid system

- MFCC Inputs
- CD
- Phone Outputs
- Hidden units
- (39*9=351)
- 2000
- 3–8 hidden layers
- 12000
- MFCC Inputs
- Hidden units
- 2000
- (39*9=351)
DNNs — what’s new?

- Training multi-hidden layers directly with gradient descent is difficult — sensitive to initialisation, gradients can be very small after propagating back through several layers.

**Unsupervised pretraining**
- Train a stacked restricted Boltzmann machine generative model (unsupervised), then finetune with backprop
- Contrastive divergence training

**Layer-by-layer training**
- Successively train deeper networks, each time replacing output layer with hidden layer and new output layer

- Many hidden layers
  - GPUs provide the computational power
- Wide output layer (context dependent phone classes)
  - GPUs provide the computational power

(Hinton et al 2012)
After it has been discriminatively fine-tuned, a DNN outputs probabilities of the form $p(HMMstate|AcousticInput)$. But to compute a Viterbi alignment or to run the forward-backward algorithm within the HMM framework, we require the likelihood $p(AcousticInput|HMMstate)$. The posterior probabilities that the DNN outputs can be converted into the scaled likelihood by dividing them by the frequencies of the HMM states in the forced alignment that is used for fine-tuning the DNN [9].

All of the likelihoods produced in this way are scaled by the same unknown factor of $p(AcousticInput)$, but this has no effect on the alignment. Although this conversion appears to have little effect on some recognition tasks, it can be important for tasks where training labels are highly unbalanced (e.g., with many frames of silences).

**Phonetic Classification and Recognition on TIMIT**

The TIMIT data set provides a simple and convenient way of testing new approaches to speech recognition. The training set is small enough to make it feasible to try many variations of a new method and many existing techniques have already been benchmarked on the core test set, so it is easy to see if a new approach is promising by comparing it with existing techniques that have been implemented by their proponents [23]. Experience has shown that performance improvements on TIMIT do not necessarily translate into performance improvements on large vocabulary tasks with less controlled recording conditions and much more training data. Nevertheless, TIMIT provides a good starting point for developing a new approach, especially one that requires a challenging amount of computation.

Mohamed et al. [12] showed that a DBN-DNN acoustic model outperformed the best published recognition results on TIMIT at about the same time as Sainath et al. [23] achieved a similar improvement on TIMIT by applying state-of-the-art techniques developed for large vocabulary recognition. Subsequent work combined the two approaches by using state-of-the-art, DT speaker-dependent features as input to the DBN-DNN [24], but this produced little further improvement, probably because the hidden layers of the DBN-DNN were already doing quite a good job of progressively eliminating speaker differences [25].

The DBN-DNNs that worked best on the TIMIT data formed the starting point for subsequent experiments on much more challenging large vocabulary tasks that were too computationally intensive to allow extensive exploration of variations in the architecture of the neural network, the representation of the acoustic input, or the training procedure. For simplicity, all hidden layers always had the same size, but even with this constraint it was impossible to train all possible combinations of number of hidden layers [1, 2, 3, 4, 5, 6, 7, 8], number of units per layer [512, 1,024, 2,048, 3,072], and number of frames of acoustic data in the input layer [7, 11, 15, 17, 27, 37]. Fortunately, the performance of the networks on the TIMIT core test set was fairly insensitive to the precise details of the architecture and the results in [13] suggest that any combination of the numbers in boldface probably has an error rate within about 2% of the very best combination. This

---

**Unsupervised pretraining**

Hinton et al. (2012)
Train a ‘baseline’ three state monophone HMM/GMM system (61 phones, 3 state HMMs) and Viterbi align to provide DNN training targets (time state alignment)

The HMM/DNN system uses the same set of states as the HMM/GMM system — DNN has 183 (61*3) outputs

Hidden layers — many experiments, exact sizes not highly critical
  - 3–8 hidden layers
  - 1024–3072 units per hidden layer

Multiple hidden layers always work better than one hidden layer

Pretraining always results in lower error rates

Best systems have lower phone error rate than best HMM/GMM systems (using state-of-the-art techniques such as discriminative training, speaker adaptive training)
Acoustic features for NN acoustic models

- GMMs: filter bank features (spectral domain) not used as they are strongly correlated with each other – would either require
  - full covariance matrix Gaussians
  - many diagonal covariance Gaussians
- DNNs do not require the components of the feature vector to be uncorrelated
  - Can directly use multiple frames of input context (this has been done in NN/HMM systems since 1990!)
  - Can potentially use feature vectors with correlated components (e.g. filter banks)
- Experiments indicate that filter bank features result in greater accuracy than MFCCs
Continuous features. A very important feature of neural networks is their "distributed representation" of the input, i.e., many neurons in the hidden layer are active simultaneously to represent each input vector. This makes neural networks good for modeling the mixture of latent causes that are active simultaneously. A recently introduced visualization method called "t-SNE" was used for producing 2-D embeddings of the input vectors or the hidden activity vectors. t-SNE produces 2-D embeddings in which points that are close in the high-dimensional vector space are likely to use latent variables that are significantly different patterns can occur in another.

Table 1 compares the Phone error rates (PER) of a shallow network with one layer of 2048 hidden units per layer) as in figure 2. The performance of fbank features is about 1% better than MFCCs (using 17 frames as input and 8 hidden layers plus a softmax output layer were used for both systems). A recently introduced visualization method called "t-SNE" [9] was used for producing 2-D embeddings of the input vectors. Regarding the comparisons here, we need to be aware that the data was generated by 15 speakers each speaking one sentence. Table 1 shows that the PER of a shallow and a deep network.

The second key idea of DBNs is "being deep." Deep acoustic models require uncorrelated data so we compared the PER of the best performing DBNs trained with MFCCs (using 17 frames as input and 3072 hidden units per layer) and the best performing DBNs trained using a mixture of diagonal covariance Gaussians. DBNs do not work with large hidden layers because it is exponentially expensive to compute the derivative of the log probability of the training data. Nevertheless, each layer can be trained efficiently using an approximate training procedure called "contrastive divergence" [8].

To understand this result we need to visualize the input vectors. One of the major motivations for generative training is the belief that the discriminations we want to perform are more directly related to the underlying causes of the acoustic data than to the individual sub-bands; each piece of data has only a single latent cause. On the other hand, a model that explains the data using multiple causes only requires other hand, a model that explains the data using multiple causes only. Similar behavior is observed in the number of trainable parameters, which are exponentially more compact than GMMs. Suppose, for example, that the sub-bands are active simultaneously to represent each input vector. This makes neural networks good for modeling the mixture of latent causes that are active simultaneously. A recently introduced visualization method called "t-SNE" was used for producing 2-D embeddings of the input vectors or the hidden activity vectors. t-SNE produces 2-D embeddings in which points that are close in the high-dimensional vector space are likely to use latent variables that are significantly different patterns can occur in another.

Table 1

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of layers</th>
<th>Features</th>
<th>Phone error rate (PER)</th>
</tr>
</thead>
<tbody>
<tr>
<td>mfcc-hid-3072-16fr-core</td>
<td>8</td>
<td>dev</td>
<td>23.6%</td>
</tr>
<tr>
<td>mfcc-hid-3072-16fr-dev</td>
<td>8</td>
<td>core</td>
<td>23.8%</td>
</tr>
<tr>
<td>fbank-hid-2048-15fr-core</td>
<td>5</td>
<td>dev</td>
<td>24.5%</td>
</tr>
<tr>
<td>fbank-hid-2048-15fr-dev</td>
<td>5</td>
<td>core</td>
<td>23.8%</td>
</tr>
</tbody>
</table>

(Mohamed et al (2012))

(TIMIT phone error rates: effect of depth and feature type)

State-of-the-art ASR systems do not use fbank coefficients as the input. One of the major motivations for generative training is the belief that the discriminations we want to perform are more directly related to the underlying causes of the data than to the individual sub-bands; each piece of data has only a single latent cause. On the other hand, a model that explains the data using multiple causes only. Similar behavior is observed in the number of trainable parameters, which are exponentially more compact than GMMs. Suppose, for example, that the sub-bands are active simultaneously to represent each input vector. This makes neural networks good for modeling the mixture of latent causes that are active simultaneously.
Visualising neural networks

- How to visualise NN layers? “t-SNE” (stochastic neighbour embedding using t-distribution) projects high dimension vectors (e.g. the values of all the units in a layer) into 2 dimensions.

- t-SNE projection aims to keep points that are close in high dimensions close in 2 dimensions by comparing distributions over pairwise distances between the high dimensional and 2 dimensional spaces – the optimisation is over the positions of points in the 2-d space.
Feature vector (input layer): t-SNE visualisation

MFCC
(Mohamed et al (2012))
Visualisation of 2 utterances (cross and circle) spoken by 6 speakers (colours)
MFCCs are more scattered than FBANK
FBANK has more local structure than MFCCs
are also close in the 2-D space. It starts by converting the pairwise
distances, \(d_{ij}\) in the high-dimensional space to ... their higher dimensionality, we consider dct features,
which are the same as fbank features except that they are trans-

![t-SNE 2-D map of the 1st layer of the fine-tuned hidden network](image1)

**MFCC**
(Mohamed et al (2012))
Visualisation of 2 utterances (cross and circle) spoken by 6
speakers (colours)
Hidden layer vectors start to align more between speakers for
FBANK
Eighth hidden layer: t-SNE visualisation

MFCC

(Mohamed et al (2012))

Visualisation of 2 utterances (cross and circle) spoken by 6 speakers (colours)

In the final hidden layer, the hidden layer outputs for the same phone are well-aligned across speakers for both MFCC and FBANK – but stronger for FBANK
Visualising neural networks

- How to visualise NN layers? “t-SNE” (stochastic neighbour embedding using t-distribution) projects high dimension vectors (e.g. the values of all the units in a layer) into 2 dimensions.
- t-SNE projection aims to keep points that are close in high dimensions close in 2 dimensions by comparing distributions over pairwise distances between the high dimensional and 2 dimensional spaces – the optimisation is over the positions of points in the 2-d space.

Are the differences due to FBANK being higher dimension $(41 \times 3 = 123)$ than MFCC $(13 \times 3 = 39)$?

- NO!
- Using higher dimension MFCCs, or just adding noise to MFCCs results in higher error rate.
- Why? – In FBANK the useful information is distributed over all the features; in MFCC it is concentrated in the first few.
Example: hybrid HMM/DNN large vocabulary conversational speech recognition (Switchboard)

- Recognition of American English conversational telephone speech (Switchboard)
- Baseline context-dependent HMM/GMM system
  - 9,304 tied states
  - Discriminatively trained (BMMI — similar to MPE)
  - 39-dimension PLP (+ derivatives) features
  - Trained on 309 hours of speech
- Hybrid HMM/DNN system
  - Context-dependent — 9304 output units obtained from Viterbi alignment of HMM/GMM system
  - 7 hidden layers, 2048 units per layer
- DNN-based system results in significant word error rate reduction compared with GMM-based system
- Pretraining not necessary on larger tasks (empirical result)
DNN vs GMM on large vocabulary tasks (Experiments from 2012)

[TABLE 3] A COMPARISON OF THE PERCENTAGE WERs USING DNN-HMMs AND GMM-HMMs ON FIVE DIFFERENT LARGE VOCABULARY TASKS.

<table>
<thead>
<tr>
<th>TASK</th>
<th>HOURS OF TRAINING DATA</th>
<th>DNN-HMM</th>
<th>GMM-HMM WITH SAME DATA</th>
<th>GMM-HMM WITH MORE DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWITCHBOARD (TEST SET 1)</td>
<td>309</td>
<td>18.5</td>
<td>27.4</td>
<td>18.6 (2,000 H)</td>
</tr>
<tr>
<td>SWITCHBOARD (TEST SET 2)</td>
<td>309</td>
<td>16.1</td>
<td>23.6</td>
<td>17.1 (2,000 H)</td>
</tr>
<tr>
<td>ENGLISH BROADCAST NEWS</td>
<td>50</td>
<td>17.5</td>
<td>18.8</td>
<td></td>
</tr>
<tr>
<td>BING VOICE SEARCH (SENTENCE ERROR RATES)</td>
<td>24</td>
<td>30.4</td>
<td>36.2</td>
<td></td>
</tr>
<tr>
<td>GOOGLE VOICE INPUT</td>
<td>5,870</td>
<td>12.3</td>
<td></td>
<td>16.0 (&gt;&gt; 5,870 H)</td>
</tr>
<tr>
<td>YOUTUBE</td>
<td>1,400</td>
<td>47.6</td>
<td>52.3</td>
<td></td>
</tr>
</tbody>
</table>

(Hinton et al (2012))
Neural Network Features
Tandem features (posteriorgrams)

- Use NN probability estimates as an additional input feature stream in an HMM/GMM system — (Tandem features (i.e. NN + acoustics), posteriorgrams)

- Advantages of tandem features
  - can be estimated using a large amount of temporal context (e.g. up to $\pm 25$ frames)
  - encode phone discrimination information
  - only weakly correlated with PLP or MFCC features

- Tandem features: reduce dimensionality of NN outputs using PCA, then concatenate with acoustic features (e.g. MFCCs)
  - PCA also decorrelates feature vector components – important for GMM-based systems
recognition systems (SRSs), particularly in the context of the conversational telephone speech recognition task. This ultimately would require both a revamping of acoustical feature extraction and a fresh look at the incorporation of these features into statistical models representing speech. So far, much of our effort has gone towards the design of new features and experimentation with their incorporation in a modern speech-to-text system. The new features have already provided significant improvements in such a system in the 2004 NIST evaluation of recognizers of conversational telephone speech. The development of statistical models to best incorporate the long time features is being explored, but development is still in its early stages.

BACKGROUND

Mainstream speech recognition systems typically use a signal representation derived from a cepstral transformation of a short-term spectral envelope. This dependence on the spectral envelope for speech sound discrimination dates back to the 1950s, as described in [11]. In turn, this style of analysis can be traced back to the 1930s vocoder experiments of Homer Dudley [14]. Perhaps more fundamentally, many speech scientists have observed the relationship between the spectral components of speech sounds and their phonetic identity. They have further characterized these sounds by their correspondence to the state of the speech articulators and the resulting resonances (formants). By this view, one should use pattern recognition techniques to classify new instances of speech sounds based on their proximity in some spectral (or cepstral) space to speech sounds collected for training the system. Modern statistical speech recognition systems are fundamentally elaborations on this principle; individual training examples are not used directly for calculating distances but rather are used to train models that represent statistical distributions. The Markov chains that are at the heart of these models represent the temporal aspect of speech sounds and can accommodate differing durations for particular instances. The overall structure provides a consistent mathematical framework that can incorporate powerful learning methods such as maximum likelihood training using expectation maximization [12]. Systems using short-term cepstra for acoustic features and first-order Markov chains for the acoustic modeling have been successful both in the laboratory and in numerous applications, ranging from cell phone voice dialing to dialog systems for use in call centers.

Despite these successes, there are still significant limitations to speech recognition performance, particularly for conversational speech and/or for speech with significant acoustic degradations from noise or reverberation. For this reason, we have proposed methods that incorporate different (and larger) analysis windows, which will be described below. We note in passing that we and many others have already taken advantage of processing techniques that incorporate information over long time ranges, for instance for normalization (by cepstral mean subtraction [2] or relative spectral analysis (RASTA) [18]). We also have proposed features that are based on speech sound class posterior probabilities, which have good properties for both classification and stream combination.

TEMPORAL REPRESENTATIONS FOR EARS

Our goal is to replace (or augment) the current notion of a spectral-energy-based vector at time \( t \) with variables based on

![Diagram showing tandem features](image)

Morgan et al (2005)
Bottleneck features

Grezl and Fousek (2008)

- Use a “bottleneck” hidden layer to provide features for a HMM/GMM system
- Decorrelate the hidden layer using PCA (or similar)
Experimental comparison of tandem and bottleneck features

Results on a Mandarin broadcast news transcription task, using an HMM/GMM system

Explores many different acoustic features for the NN

Posteriorgram/bottleneck features alone (top)

Concatenating NN features with MFCCs (bottom)
Autoencoders

- An autoencoder is a neural network trained to map its input into a distributed representation from which the input can be reconstructed.
- Example: single hidden layer network, with an output the same dimension as the input, trained to reproduce the input using squared error cost function.

\[ E = -\frac{1}{2} ||y - x||^2 \]

- \( y \): \( d \) dimension outputs
- \( x \): \( d \) dimension inputs
- learned representation
Autoencoder Bottleneck (AE-BN) Features

- First train a “usual” DNN classifying acoustic input into 384 HMM states
- Then train an autoencoder that maps the predicted output vector to the target output vector
- Use the bottleneck hidden layer in the autoencoder as features for a GMM/HMM system
Results using Autoencoder Bottleneck (AE-BN) Features

<table>
<thead>
<tr>
<th>LVCSR STAGE</th>
<th>50 H GMM-HMM BASELINE</th>
<th>50 H AE-BN</th>
<th>430 H GMM/HMM BASELINE</th>
<th>430 H AE-BN</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSA</td>
<td>24.8</td>
<td>20.6</td>
<td>20.2</td>
<td>17.6</td>
</tr>
<tr>
<td>+fBMMI</td>
<td>20.7</td>
<td>19.0</td>
<td>17.7</td>
<td>16.6</td>
</tr>
<tr>
<td>+BMMI</td>
<td>19.6</td>
<td>18.1</td>
<td>16.5</td>
<td>15.8</td>
</tr>
<tr>
<td>+MLLR</td>
<td>18.8</td>
<td>17.5</td>
<td>16.0</td>
<td>15.5</td>
</tr>
<tr>
<td>MODEL COMBINATION</td>
<td>16.4</td>
<td></td>
<td>15.0</td>
<td></td>
</tr>
</tbody>
</table>

Hinton et al (2012)
Summary

- DNN/HMM systems (hybrid systems) give a significant improvement over GMM/HMM systems.
- Compared with 1990s NN/HMM systems, DNN/HMM systems:
  - model context-dependent tied states with a much wider output layer.
  - are deeper – more hidden layers.
  - can use correlated features (e.g. FBANK).
- DNN features obtained from output layer (posteriorgram) or hidden layer (bottleneck features) give a significant reduction in WER when appended to acoustic features (e.g. MFCCs).
Reading


