MPhil in Machine Learning and Machine Intelligence

Module MLMI2: Speech Recognition

L10: Advanced Neural Network Acoustic Models

Phil Woodland
pcw@eng.cam.ac.uk

Michaelmas 2021

Cambridge University Engineering Department

Introduction

This lecture discusses the use of more advanced/structured neural network acoustic models in the context of ANN-HMM systems.

We will concentrate on techniques that offer **better modelling of the time and frequency domains** than the feed-forward fully connected DNNs and basic RNNs discussed so far.

We will cover the following topics

- Review of DNN/RNN-HMM acoustic models
- Limitations of feed-forward fully-connected DNNs and RNNs
- Time Delay Neural Networks (TDNNs)
- Convolutional Neural Networks (CNNs)
- Increasing network depth
- **Gated** recurrent networks
  - Long Short-Term Memory (LSTM) models
  - LSTM Extensions/Alternatives
- Grid recurrent networks
- Combined models

Note: still assuming a **hybrid** approach to speech recognition and improving the acoustic model. In this lecture we are not discussing **end-to-end trainable** neural network speech recognition approaches.

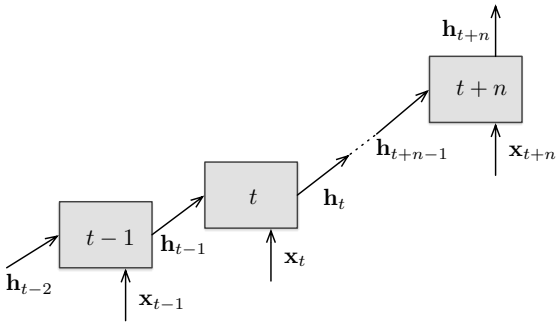
Review of DNN/RNN-HMM Acoustic Models

Feed-forward fully connected DNN:

- Input is typically a set of frames **context-window**
- Set of fully-connected (FC) layers each with a **non-linear activation function**
- Outputs are **context-dependent phone state** (or CI phone states)
- Estimate the posterior probability of output using **softmax** activation function : convert to a **scaled likelihood** by dividing by the prior
- Use scaled likelihood **in place of GMM** output probability in DNN-HMM structure

Recurrent NN:

- Recurrent layer computed using hidden input from previous time and current frame input
- Same outputs and use in ANN-HMM structure
- **Unfold** recurrent structure to feed-forward & limit depth (20 frames)
- Can add extra feed-forward FC layers after recurrent layer(s)
- Train using **truncated back propagation through time** to estimate gradients



Limitations of feed-forward fully-connected DNNs and RNNs

Feed-forward fully-connected DNNs are powerful structures.

- Each hidden layer performs a non-linear transform that can be thought of **extracting features**
- However the features extracted do not have **time/frequency invariance**
- Speech features should often have time invariance (i.e. speech sounds can be shifted in time)
- Some frequency invariance is useful for speech
- Often **depth of network** limited
- Fixed input time widow allows limited time evolution to be explicitly learned

RNNs can learn dependencies through time but:

- In practice suffer from **vanishing gradients**
- ReLU RNNs suffer less from vanishing gradients than sigmoids
- Still limited to the time extent over which can practically learn
- **Gated structures with explicit memory** can (more easily) learn long-term dependencies

Consider the type of network architectures that can address these issues and combinations of network types.

Time Delay Neural Networks (TDNNs)

- A TDNN consists of identical (i.e. shared) fully-connected (FC) layers **replicated at different time steps** (i.e.different positions in a normal DNN context window)
- Replicated FC layer allows features to be computed with time-shift invariance and reduces number of parameters for a certain sized temporal context window
- Next layer also takes as input a stack of different time steps of the preceding layer (can also be shared across time steps).
- Initial layers thus learn to detect features within fairly narrow temporal contexts
- Further layers operate on a much larger temporal context.
- Originally introduced by Waibel et al (1989) for classification of segmented consonant-vowels.
- Was one inspiration for convolutional neural networks (CNNs).

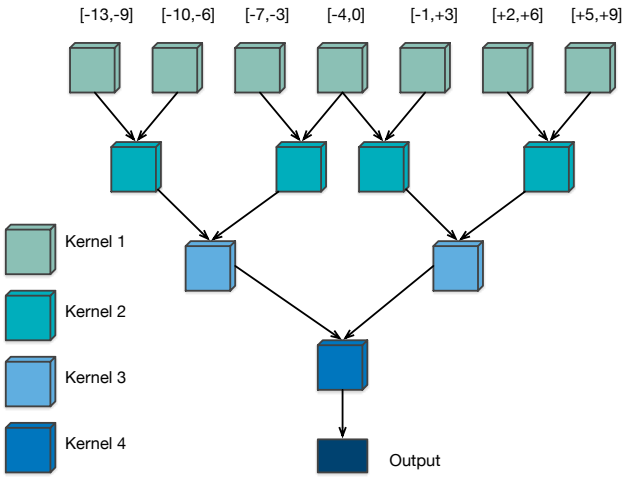
More recently TDNNs revisited in the context of LVCSR & hybrid ANN-HMM systems.

- Original TDNN uses shifts of one frame in time for the FC layers and no extra context frames (expensive for large overall temporal contexts).
- Sub-sampled TDNN (Peddinti et al, 2015) has larger frame shifts and context widows
- Allows an overall very wide context window
- Smaller number of parameters for wide temporal context
- Effective with less training data than for same temporal context with std DNN

Sub-Sampled TDNNs

Subsampled TDNN structure shown (input at top).

- Moves first FC layer across window $t \in [-13, 9]$ with temporal context of 5 frames at shifts of 3 frames
- Total input context split into 7 time-bins
- Followed by a binary tree combination of the outputs of the layer.
- Each Kernel could be a standard FC layer or could be more complex to increase network depth



- For all TDNNs, trained using error back-propagation to compute gradients
  - Gradients are accumulated over all instantiations of FC layers
  - Accumulated gradients normalised before SGD updates are computed
  - Same idea applied to other cases when training with **shared weights**
- Widely used for speech (in part due to **Kaldi** implementation/recipe)

Convolutional Neural Networks (CNNs)

- CNNs have been widely used for processing images: set of 2-D filters are learned
- In a CNN layer a set of 2-D filters is convolved with the input which results in **multiple output-maps**, one per filter
- Followed by element-wise activation function, such as the sigmoid or $\sigma(\cdot)$ function
- For speech, CNNs use **2-D convolutional filters** applied to time context window of e.g. Mel-filterbank outputs

Output of a convolutional layer is for input time-frequency "image" with points $x_{i,j}$

$$h_{i,j,k} = \sigma\Big(\sum_{l=0}^{T-1}\sum_{m=0}^{F-1} x_{i+l,j+m} \cdot w_{l,m,k}\Big), \quad i = 1 \ldots L-T; \;\; j = 1 \ldots M-F; \;\; k = 1, \ldots, K$$

where L is dimensionality of time-axis, M is the dimensionality of frequency-axis, $T \times F$ is the size of the filters, k is index of filter and K is the number of filters.

Generally followed by a **pooling** operation which "summarises" patches in each output map

- Computes average (average-pooling) or maximum value (max-pooling).
- Allows for some invariance to shifts in the location of a feature.

In above, same filter applied across entire input space: **full weight sharing** (FWS).

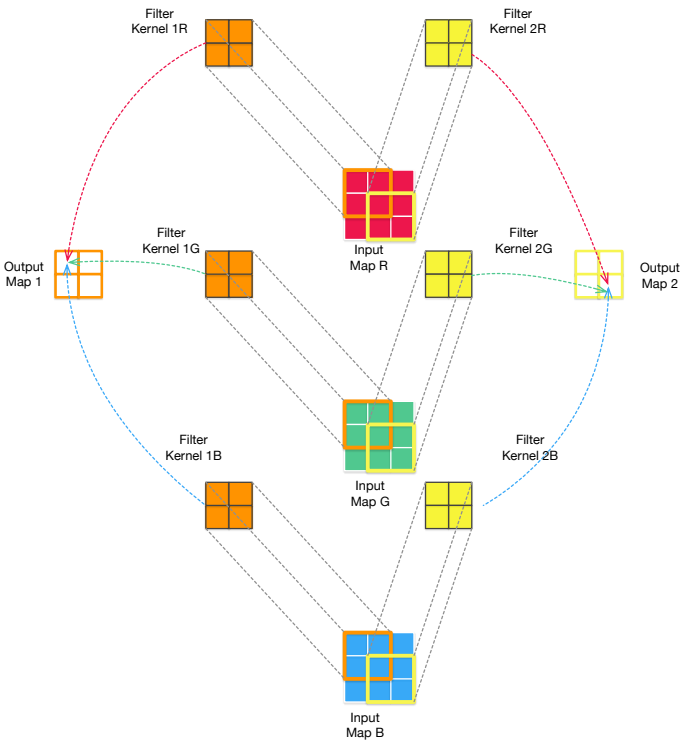
- Assumes that features occurs across entire input space (ok for time, but frequency?)
- **limited weight sharing** (LWS) shares same filter only where output values are pooled.
- possible to have intermediates between these two extremes for speech

Note that CNN layers are **typically followed by fully-connected layers** before classification.

CNN Illustration

Operation of one layer of a CNN with a small 3x3 feature map and 2 filters is shown with 2x2 output maps.

- Image based CNNs normally use 3 channels: RGB
- Separate filters for each channel
- For speech, separate channels are for static, Δ, & $\Delta\Delta$ input features.
- When separate channels are used the filter outputs are added together in the output filter map
- If want to maintain dimensionality of input would need to use **padding** to extend input "image".

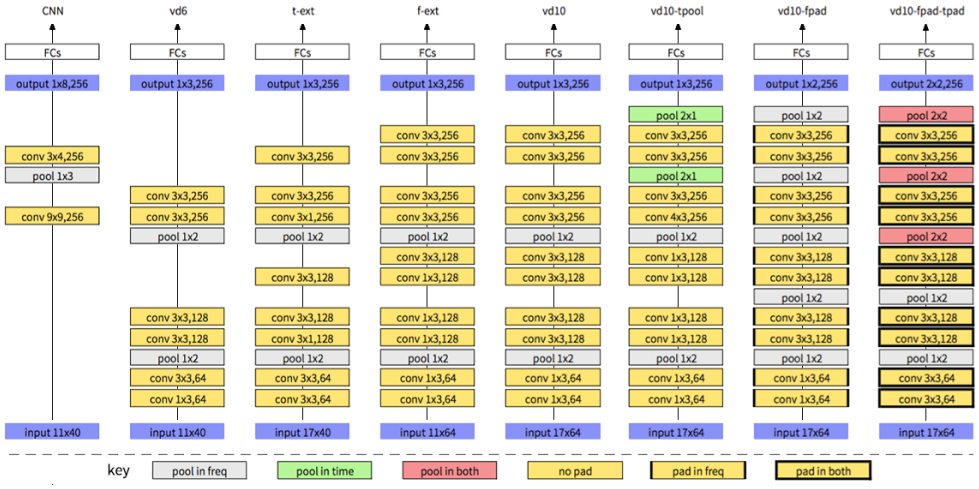


Note that typically for speech recognition two CNN layers are used.

- 9x9 (TxF) first layer filters
- 1x3 pooling (i.e. freq. only)
- 3x4 second CNN layer filters

(Very) Deep CNNs

- Originally CNN 2-D filters covered larger areas of input “image”
- One trend in image processing towards much deeper networks of much smaller filters. These are often known as VGG networks (after Oxford group).
- Note also need to consider possibility of padding at the edges
- Many architectures possible: some shown below (from Qiang & Woodland, 2016).



- Models of this form have been used for **bottleneck feature extraction** for stacked DNNs (see next lecture) as well as in hybrid ANN-HMM models.

Increasing network depth

Network depth is viewed as important in creating networks with more modelling power.

One key issue is the ability to estimate the parameters of very deep networks.

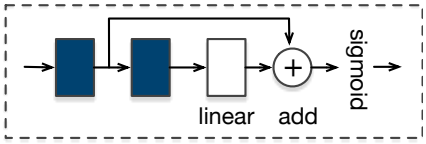
This can be partially over come by the related **Residual** connections and **Highway** models.

Residual Connections (**ResNet**)

- Identity mapping from the output of initial layers to the input of later layer (skipping layers)
- For standard hidden layer transformation $\mathbf{y} = H(\mathbf{x}, \mathbf{W}_H)$, ResNet connection is

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W}_H) + \mathbf{x}$$

- Also possible to skip more than one layer
- back-propagation of the gradients more effective
- minimum effective depth is reduced
- training is more effective/faster/better



- One form of ResNet kernel

Highway Connections. This replaces a standard hidden layer by a highway connection alternative with the extra transform $T(\mathbf{x}, \mathbf{W}_T)$

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W}_H).T(\mathbf{x}, \mathbf{W}_T) + \mathbf{x}.(1 - T(\mathbf{x}, \mathbf{W}_T))$$

Similar aim to ResNet: depending on the value of $T(\cdot)$ can either represent the standard hidden layer transform, the input vector or some combination. Example of **gating**.

Long Short-Term Memory (LSTM) models

- LSTMs consist of a set of recurrently connected subnetworks, called memory blocks
- Each memory block contains memory cells to store temporal cell state, $\mathbf{c}_t$, as well as three multiplicative gate units to control information flow.
- **Input Gate** controls information passed from input activations into memory cells, $\mathbf{i}_t$
- **Output Gate** controls information passed from the memory cell to rest of network, $\mathbf{o}_t$
- **Forget Gate** adaptively resets the memory of the cell, $\mathbf{f}_t$

Controlled by the following equations, with input $\mathbf{x}_t$ and hidden state $\mathbf{h}_t$:

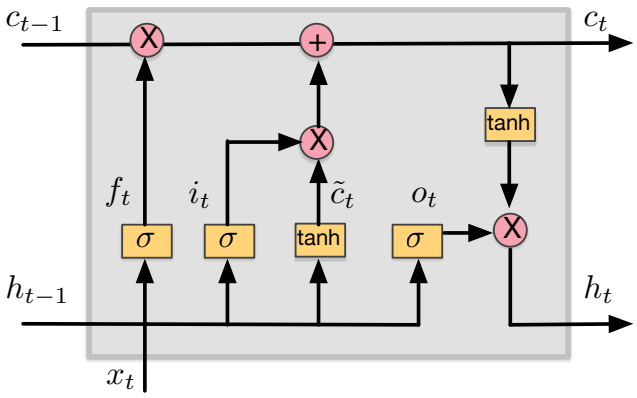
$$
\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i.\,[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_f.\,[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c.\,[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o.\,[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned}
$$

Here $\tilde{\mathbf{c}}_t$ is known as the candidate cell value; $\mathbf{W}_f$, $\mathbf{W}_i$, $\mathbf{W}_o$ are weight matrices associated with the gates, $\odot$ represents element-wise multiplication.

The computation steps and description is shown next

LSTM Computation

- Compute input gate, forget gate, and current cell value candidate for memory cell.
- Update memory cell value. Previous memory cell value re-weighted using forget gate, & added to current cell candidate weighted by input gate.
- Updated memory cell value transformed with `tanh` function & then re-weighted using output gate value.



LSTMs, like standard RNNs can still be **trained by truncated BPTT** and SGD.

Note that the LSTM has about 4x as many parameters as a simple RNN with the same hidden layer dimensionality

- Structure is computationally expensive due to increase in parameters
  - Interest in reducing parameters/computation while retaining model accuracy
- Good at solving vanishing gradient problem due to explicit memory
- Can give very good performance in speech recognition
- Also used as basis of some end-to-end trainable sequence-to-sequence models

LSTM Enhancements/Alternatives

A single layer LSTM has been described above:

- multiple layers of LSTMs improve performance
- bidirectional layers (backwards layer normally increases latency)

One common modification to the LSTM is the use of **"peephole"** connections

- Allow the gates to depend on the cell state as well as the previous hidden state and input

This modifies the basic LSTM equations for the input gate, forget gate and output gate as follows:

$$\begin{aligned}\mathbf{i}_t &= \sigma(\mathbf{W}_i.[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{V}_i \odot \mathbf{c}_{t-1} + \mathbf{b}_i)\\ \mathbf{f}_t &= \sigma(\mathbf{W}_f.[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{V}_f \odot \mathbf{c}_{t-1} + \mathbf{b}_f)\\ \mathbf{o}_t &= \sigma(\mathbf{W}_o.[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{V}_o \odot \mathbf{c}_t + \mathbf{b}_o)\end{aligned}$$

where the extra $\mathbf{V}$ vectors for the peephole connections are shown.

A simpler alternative gated model is the **Gated Recurrent Unit (GRU) model**. This has a number of changes including:

- Merges cell state and hidden state
- Uses update gate: merges forget/input gates: $1 - \mathbf{i}_t$ used instead of forget gate

LSTM Computation Reduction

One key issue of LSTMs for ASR is computational load due to large number of parameters.

One popular architecture modification for speech is the **Projected LSTM (LSTMP)**.

- recurrent connection $\mathbf{h}_{t-1}$ replaced by a lower dimensional version computed by another recurrent connection layer & weight matrix: can be viewed as a **low-rank approximation**.
- LSTM weight matrices are now smaller & significant savings in computation/parameters
- $\mathbf{h}_t$ is still retained for the network output (connection to output layer)

Another possibility is **Semi-tied LSTM Units**.

- Computes a common "virtual unit" $\mathbf{e}_t = \sigma(\mathbf{W}.[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}$
- and then the various values uses this vale to compute the individual gates

$$\begin{aligned}&\mathbf{i}_t = \sigma_{\boldsymbol{\eta}_i,\gamma_i}(\mathbf{e}_t + \mathbf{V} \odot \mathbf{c}_{t-1}),\ \ \mathbf{f}_t = \sigma_{\boldsymbol{\eta}_f,\gamma_f}(\mathbf{e}_t + \mathbf{V} \odot \mathbf{c}_{t-1}),\ \ \mathbf{o}_t = \sigma_{\boldsymbol{\eta}_o,\gamma_o}(\mathbf{e}_t + \mathbf{V} \odot \mathbf{c}_t)\\ &\tilde{\mathbf{c}}_t = \tanh_{\boldsymbol{\eta}_c,\gamma_c}(\mathbf{e}_t)\end{aligned}$$

  where the $\sigma_{\boldsymbol{\eta},\gamma}$ are specific **parameterised activation functions** that have extra parameters to keep the units distinct.
- Been shown that these can produce similar WERs to LSTMs but with roughly 25% of the computation/storage in the recurrent LSTM units

Various other types architecture changes have been proposed to simplify LSTM structure & paramters and hence reduce computation.

Are LSTMs Needed for Acoustic Modelling?

LSTMs are more effective than simple RNNs for many tasks including acoustic models. This is in part due to the ability to learn longer term dependencies.

However there is a great deal of extra computation and storage required, and is isn't clear if this is needed for the acoustic model in a hybrid ANN-HMM setup. Could the same effects be realised more simply?

The **Higher Order Recurrent Neural Network (HORNN)** model for ReLU models uses in the simplest case:

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}_1\mathbf{h}_{t-1} + \mathbf{U}_n\mathbf{h}_{t-n} + \mathbf{b})$$

which has separate weighted connections from the previous time step $t-1$ and from $t-n$.

To further reduce parameters the projected form HORNNP uses

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}_{p1}\mathbf{P}\mathbf{h}_{t-1} + \mathbf{U}_{pn}\mathbf{P}\mathbf{h}_{t-n} + \mathbf{b})$$

which **factorises** $\mathbf{U}_1$ and $\mathbf{U}_n$ with $\mathbf{U}_{p1}\mathbf{P}$ and $\mathbf{U}_{pn}\mathbf{P}$.

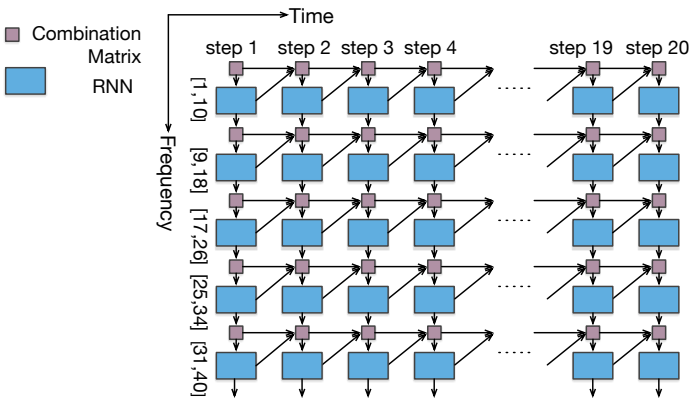
It can be shown (Zhang & Woodland, 2018) that with $n = 4$ ReLU HORNNPs can give:

- similar WERs to LSTMPs with the same hidden layer size & far fewer parameters
- lower WERs with the same overall number of parameters.

Grid recurrent networks

Interest in 2-D recurrent structures to explicitly model correlations in time and frequency.

- Normally use LSTMs but for simplicity consider simple RNN structures to show key features
- Time-Frequency (TF) LSTM and Grid LSTMs have been defined
- Unfold the structures in both time and frequency



For **TF RNN** structure to particular time step t, frequency band k, use separate recurrent matrices, $\mathbf{V}_T$ and $\mathbf{V}_F$ that operate based on "previous" time and frequency hidden state:

$$\mathbf{h}_{t,k} = \sigma\left(\mathbf{W}\mathbf{x}_{t,k} + \mathbf{V}_T\mathbf{h}_{t-1,k} + \mathbf{V}_F\mathbf{h}_{t,k-1} + \mathbf{b}\right)$$

2D-Grid-RNN has separately evolving hidden state for time, $\mathbf{h}^T_{t,k}$ and frequency $\mathbf{h}^F_{t,k}$ within the same model

$$\mathbf{h}^T_{t,k} = \sigma\left(\mathbf{W}^T\mathbf{x}_{t,k} + \mathbf{V}_T\mathbf{h}^T_{t-1,k} + \mathbf{V}_F\mathbf{h}^F_{t,k-1} + \mathbf{b}^T\right)$$

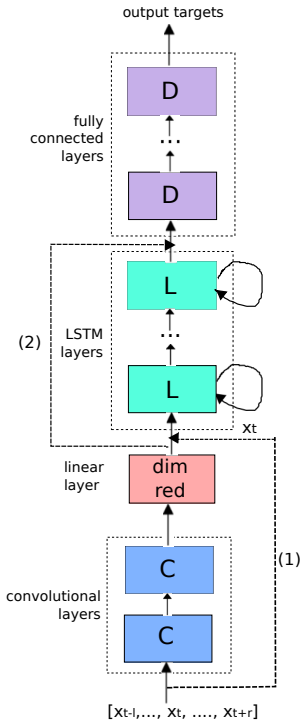
$$\mathbf{h}^F_{t,k} = \sigma\left(\mathbf{W}^F\mathbf{x}_{t,k} + \mathbf{V}_T\mathbf{h}^T_{t-1,k} + \mathbf{V}_F\mathbf{h}^F_{t,k-1} + \mathbf{b}^F\right)$$

Combined models

There are a very large number of ways that ideas presented above can be combined.

Oner popular architecture is the CLDNN (illustrated right).

- Initial CNN layers (for feature extraction)
- LSTM layers for modelling temporal evolution
- DNN FC layers for final classification
- also pass the original features to the LSTM layers directly as well
- also pass CNN features to the DNN layers



There are many other possibilities:

- Grid LSTM models have been used in place of CNNs: then combined with standard 1-D LSTMs.
- Can also combine grid models with TDNNs.

Summary

- Review of DNN-HMM and RNN-HMMs
- Limitations for modelling speech signals
- Time-Delay Neural Networks
  - Sub-sampled for of TDNNs
- Convolutional neural networks
  - 2-D convolution filters learned
  - Pooling and padding
  - limited weight sharing often used for speech
  - very deep CNNs
- Learning Deep Models: ResNet & Highway
- Long short-term memory models
  - Addition of memory and gates to recurrent structure
  - Able to better learn long-term dependencies
  - Modifications: Peepholes and GRUs
  - Computation reduction via LSTMP and semi-tied units
- Higher order RNNs and HORNNPs perform well
- Grid RNNs
- Combined models

ANN techniques for ASR rapidly evolving. New models/ideas are constantly being developed. Investigate further in encoder structures used in end-to-end trainable systems.

References

- O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn & D. Yu. “Convolutional Neural Networks for Speech Recognition.” *IEEE/ACM Trans. ASLP*, vol. 22, pp. 1533-1545, 2014.
- W. Hartmann, R. Hsiao, T. Ng, J. Ma, F. Keith, M-H. Siu. “ Improved Single System Conversational Telephone Speech Recognition with VGG Bottleneck Features.” *Proc. Interspeech*, 2017.
- S. Hochreiter & J. Schmidhuber, “Long Short-Term Memory.” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- H. Huang & B. Mak, “To improve the robustness of LSTM-RNN acoustic models using higher-order feedback from multiple histories.” *Proc. Interspeech*, 2017.
- N. Kalchbrenner, I. Danihelka, & A. Graves, “Grid long short-term memory.” *Proc. ICLR*, 2016.
- F.L. Kreyssig, C. Zhang, & P.C. Woodland, “Improved TDNNs using deep kernels and frequency dependent Grid-RNNs.” *Proc. ICASSP*, 2018.
- B. Li & T. N. Sainath. “Reducing the Computational Complexity of Two-Dimensional LSTMs.” *Proc. Interspeech*, 2017.
- J. Li, A. Mohamed, G. Zweig, & Y. Gong. “Exploring Multidimensional LSTMs for Large Vocabulary ASR.” *Proc. ICASSP*, Shanghai, 2016.
- V. Peddinti, D. Povey, & S. Khudanpur. “A Time Delay Neural Network Architecture for Efficient Modeling of Long Temporal Contexts.” *Proc. Interspeech*, Dresden, 2015.

- H. Sak, A. Senior, & F. Beaufays. “Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling”, *Proc. Interspeech*, 2014.
- T. N. Sainath, O. Vinyals, A. Senior & H. Sak. “Convolutional, Long Short-Term Memory, Fully Connected Deep Neural Networks.” *Proc. ICASSP*, 2015.
- G. Saon, Z Tuske, K. Audhkhasi, B. Kingsbury, M. Picheny & S. Thomas. “Simplified LSTMs for Speech Recognition”, *Proc. ASRU*, 2019.
- T. Sercu, C. Puhrsch, B. Kingsbury, & Y. LeCun, “Very deep multilingual convolutional neural networks for LVCSR.” *Proc. ICASSP*, 2016.
- Y. Qian & P.C. Woodland. “Very Deep Convolutional Neural Networks for Robust Speech Recognition.” Proc. IEEE Workshop on Spoken Language Processing, 2016.
- A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, & K.J. Lang. “Phoneme Recognition using Time-Delay Neural Networks.” *IEEE Trans ASSP*, vol. 37, no. 3, pp. 328–339, 1989.
- C. Zhang & P.C. Woodland, “High order recurrent neural networks for acoustic modelling” *Proc. ICASSP*, 2018
- C. Zhang & P.C. Woodland, “Semi-tied units for efficient gating in LSTM and highway networks.” *Proc. Interspeech*, 2018.