Deep Learning for Speech Processing

An NST Perspective

Natural
Speech
Technology

Edinburgh – Cambridge – Sheffield

Mark Gales

University of Cambridge

June 2016
September 2016

Natural Speech Technology (NST)

- EPSRC (UK Government) Programme Grant: collaboration

THE UNIVERSITY of EDINBURGH informatics

The University Of Sheffield.

UNIVERSITY OF CAMBRIDGE
Department of Engineering

Natural Speech Technology

  - significantly advance state-of-the-art in speech technology
  - more natural, approaching human levels of reliability, adaptability and conversational richness
  - ran from 2011 to 2016 - interesting times ...

What is Deep Learning?

From Wikipedia:

> *Deep learning is a branch of machine learning based on a set of algorithms that attempt to model high-level abstractions in data by using multiple processing layers, with complex structures or otherwise, composed of multiple non-linear transformations.*

What is Deep Learning?

Natural Speech Technology

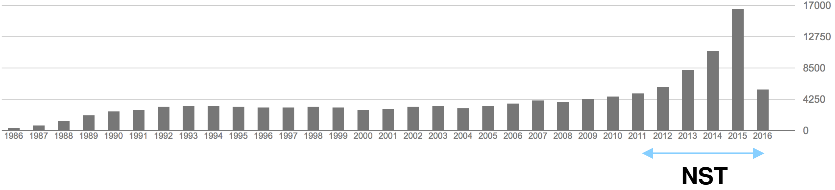
From Wikipedia:

Deep learning is a branch of machine learning based on a set of algorithms that attempt to model high-level abstractions in data by using multiple processing layers, with complex structures or otherwise, composed of multiple non-linear transformations.

For NST:

Deep learning allows both speech synthesis and speech recognition to use the same underlying, highly flexible, building blocks and adaptation techniques (and improved performance).

The Rise of Deep Learning (June 2016)



Overview

- Basic Building Blocks
  - neural network architectures
  - activation functions
- Sequence-to-Sequence Modelling
  - generative models
  - discriminative models
  - encoder-decoder models
- Speech Processing Applications
  - language modelling
  - speech recognition
  - speech synthesis
- Adaptation for Deep Learning

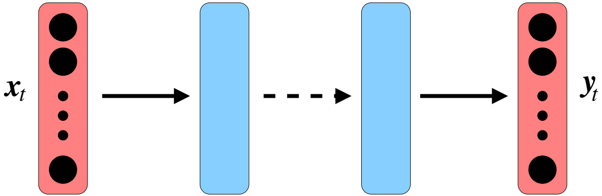
What I am Not Discussing

- History of development
- Optimisation
  - essential aspect of all systems (major issue)
- Only work from NST
  - though (of course) the talk is biased
  - indicates papers/contributions from NST in area
- Experimental Results
  - see NST publications and other papers (references at end)

- My personal opinion of DNNs ...

Basic Building Blocks

Deep Neural Networks [19]

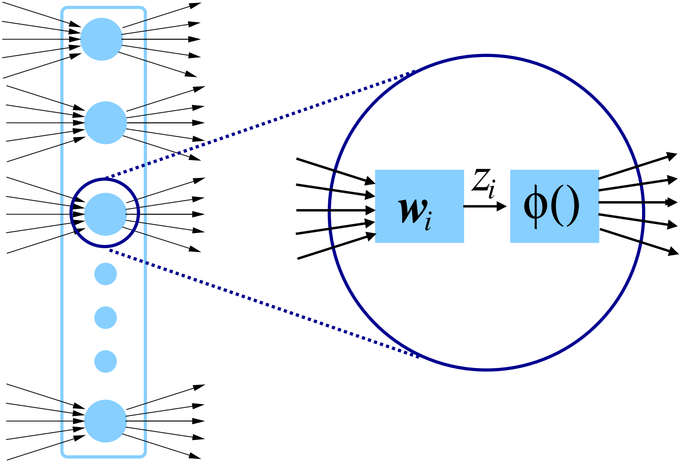


- General mapping process from input $\boldsymbol{x}_t$ to output $\boldsymbol{y}_t$

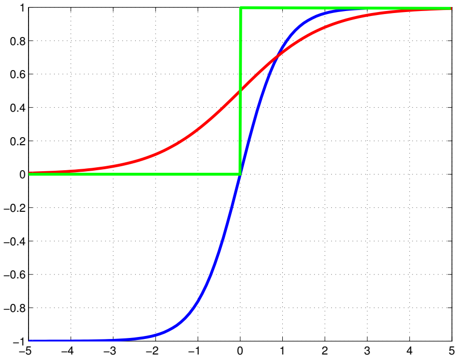
$$\boldsymbol{y}_t = \mathcal{F}(\boldsymbol{x}_t)$$

  - deep refers to number of hidden layers
- Depending on nature of outputs ($\boldsymbol{y}_t$) vary activation function
  - softmax often for classification
  - tanh or sigmoid common for hidden layers

Neural Network Layer/Node



Activation Functions



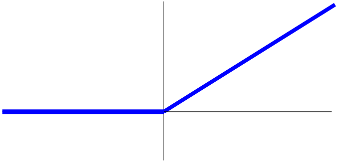
- Activation functions:
  - step function (green)
  - sigmoid function (red)
  - tanh function (blue)

- softmax, usual output layer (sum-to-one/positive) for classification

$$\phi_i(\boldsymbol{x}_t) = \frac{\exp\left(\mathbf{w}_i^{\mathrm{T}}\boldsymbol{x}_t + b_i\right)}{\sum_{j=1}^{J}\exp\left(\mathbf{w}_j^{\mathrm{T}}\boldsymbol{x}_t + b_j\right)}$$

Activation Functions - ReLUs [35, 54]

- Alternative activation function: Rectified Linear Units



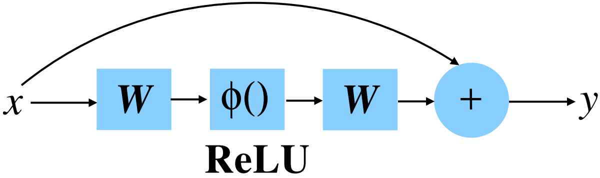
$$y_i = \max(0, z_i)$$

- Related activation function noisy ReLU:

$$y_i = \max(0, z_i + \epsilon); \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

  - efficient, no exponential/division, rapid convergence in training
- Possible to train the activation function parameters
  - e.g. gradient of slopes for ReLUs

Activation Functions - Residual Networks [17]



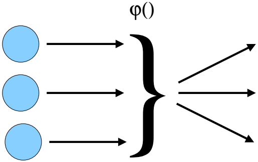
- Modify layer to model the residual

$$\boldsymbol{y}_t = \mathcal{F}(\boldsymbol{x}_t) + \boldsymbol{x}_t$$

  - allows deeper networks to be built
  - deep residual learning
- Links to highway connections

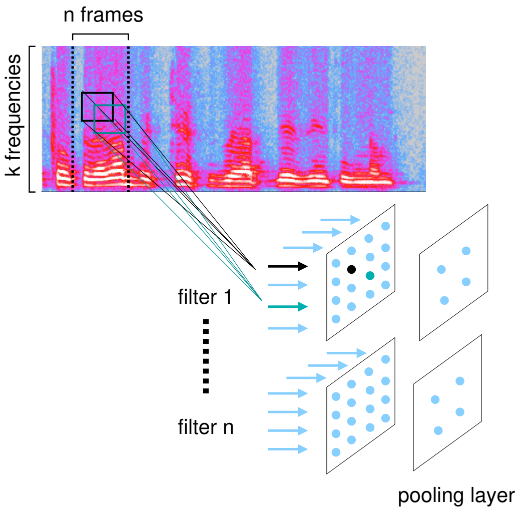
Pooling/Max-Out Functions [27, 43]

- Possible to pool the output of a set of node
  - reduces the number of weights to connect layers together

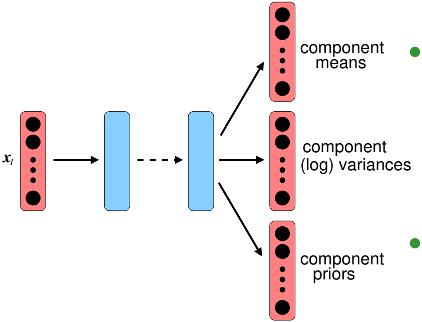


- A range of functions have been examined
  - maxout $\phi(y_1, y_2, y_3) = \max(y_1, y_2, y_3)$
  - soft-maxout $\phi(y_1, y_2, y_3) = \log(\sum_{i=1}^{3} \exp(y_i))$
  - p-norm $\phi(y_1, y_2, y_3) = (\sum_{i=1}^{3} |y_i|)^{1/p}$
- Has also been applied for unsupervised adaptation

Convolutional Neural Networks [26, 1]



Mixture Density Neural Networks [4, 52]



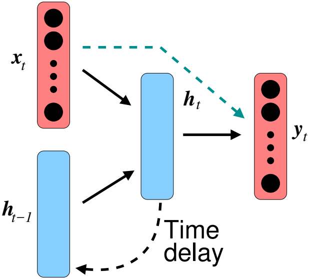
- Predict a mixture of experts
  - multiple components M
  - $\mathcal{F}_m^{(c)}(\boldsymbol{x}_t)$: prior prediction
  - $\mathcal{F}_m^{(\mu)}(\boldsymbol{x}_t)$: mean prediction
  - $\mathcal{F}_m^{(\sigma)}(\boldsymbol{x}_t)$: variance prediction
- Optimise using maximum likelihood

$$p(\boldsymbol{y}_t|\boldsymbol{x}_t) = \sum_{m=1}^{M} \mathcal{F}_m^{(c)}(\boldsymbol{x}_t)\mathcal{N}(\boldsymbol{y}_t; \mathcal{F}_m^{(\mu)}(\boldsymbol{x}_t), \mathcal{F}_m^{(\sigma)}(\boldsymbol{x}_t))$$

- Form of output influences output activation function used

Recurrent Neural Networks [38, 37]

Natural Speech Technology



- Introduce recurrent units

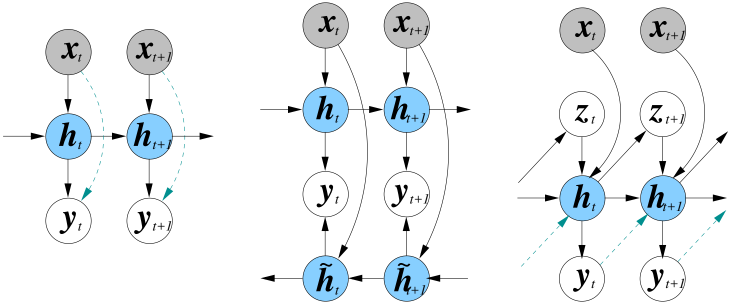
$$
\begin{aligned}
\boldsymbol{h}_t &= \mathbf{f}^{\mathrm{h}}\left(\mathbf{W}_{\mathrm{h}}^{\mathrm{f}}\boldsymbol{x}_t + \mathbf{W}_{\mathrm{h}}^{\mathrm{r}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathrm{h}}\right) \\
\boldsymbol{y}_t &= \mathbf{f}^{\mathrm{f}}\left(\mathbf{W}_{\mathrm{y}}\boldsymbol{h}_t + \mathbf{W}_{\mathrm{x}}\boldsymbol{x}_t + \boldsymbol{b}_{\mathrm{y}}\right)
\end{aligned}
$$

  - $\boldsymbol{h}_t$ history vector at time t
- Two history weight matrices
  - $\mathbf{W}_{\mathrm{h}}^{\mathrm{f}}$ forward
  - $\mathbf{W}_{\mathrm{h}}^{\mathrm{r}}$ recursion
- Uses (general) approximation (no optional i/o link)

$$
p(\boldsymbol{y}_t|\boldsymbol{x}_{1:t}) = p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{x}_{1:t-1}) \approx p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) \approx p(\boldsymbol{y}_t|\boldsymbol{h}_t)
$$

  - network has (causal) memory encoded in history vector ($\boldsymbol{h}_t$)

RNN Variants [40, 10]



RNN (+i/o link) Bi-Directional RNN Variational RNN

- Bi-directional: use complete observation sequence - non-causal
- Variational: introduce a latent variable sequence $\boldsymbol{z}_{1:T}$

$$p(\boldsymbol{y}_t|\boldsymbol{x}_{1:t}) \approx \int p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{z}_t, \boldsymbol{h}_{t-1})p(\boldsymbol{z}_t|\boldsymbol{h}_{t-1})d\boldsymbol{z}_t$$

Network Gating

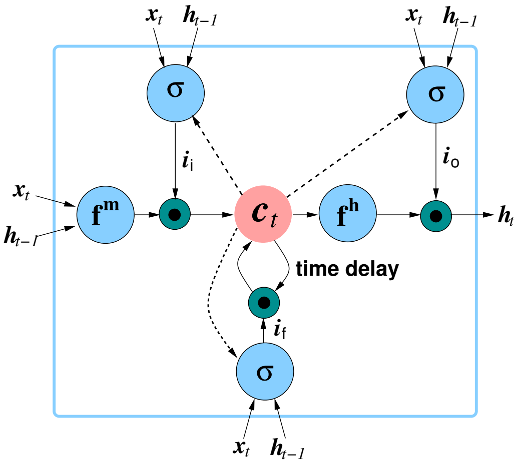
- A flexible extension to activation function is gating
  - standard form is ($\boldsymbol{\sigma}()$ sigmoid activation function)

$$\boldsymbol{i} = \boldsymbol{\sigma}(\mathbf{W}^{\mathrm{f}}\boldsymbol{x}_t + \mathbf{W}^{\mathrm{r}}\boldsymbol{h}_{t-1} + \boldsymbol{b})$$

  - vector acts a a probabilistic gate on network values

- Gating can be applied at various levels
  - features: impact of input/output features on nodes
  - time: memory of the network
  - layer: influence of a layer's activation function

Long-Short Term Memory Networks [20, 16]



Long-Short Term Memory Networks

Natural Speech Technology

- The operations can be written as (peephole config):
  - Forget gate ($\boldsymbol{i}_{\mathtt{f}}$), Input gate ($\boldsymbol{i}_{\mathtt{i}}$), Output gate ($\boldsymbol{i}_{\mathtt{o}}$)

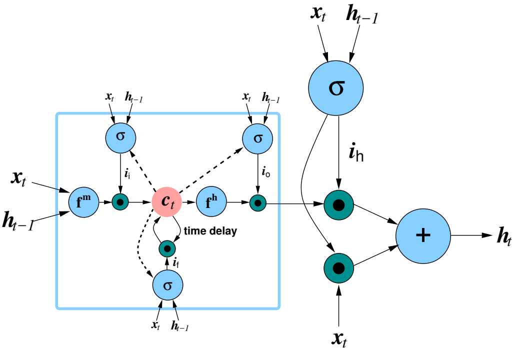
$$\begin{aligned}
\boldsymbol{i}_{\mathtt{f}} &= \boldsymbol{\sigma}(\mathbf{W}^{\mathtt{f}}_{\mathtt{f}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{f}}\boldsymbol{h}_{t-1} + \mathbf{W}^{\mathtt{m}}_{\mathtt{f}}\boldsymbol{c}_{t-1} + \boldsymbol{b}_{\mathtt{f}}) \\
\boldsymbol{i}_{\mathtt{i}} &= \boldsymbol{\sigma}(\mathbf{W}^{\mathtt{f}}_{\mathtt{i}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{i}}\boldsymbol{h}_{t-1} + \mathbf{W}^{\mathtt{m}}_{\mathtt{i}}\boldsymbol{c}_{t-1} + \boldsymbol{b}_{\mathtt{i}}) \\
\boldsymbol{i}_{\mathtt{o}} &= \boldsymbol{\sigma}(\mathbf{W}^{\mathtt{f}}_{\mathtt{o}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{o}}\boldsymbol{h}_{t-1} + \mathbf{W}^{\mathtt{m}}_{\mathtt{o}}\boldsymbol{c}_{t} + \boldsymbol{b}_{\mathtt{o}})
\end{aligned}$$

  - Memory Cell, history vector and gates are related by

$$\begin{aligned}
\boldsymbol{c}_t &= \boldsymbol{i}_{\mathtt{f}} \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_{\mathtt{i}} \odot \mathbf{f}^{\mathtt{m}}(\mathbf{W}^{\mathtt{f}}_{\mathtt{c}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{c}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathtt{c}}) \\
\boldsymbol{h}_t &= \boldsymbol{i}_{\mathtt{o}} \odot \mathbf{f}^{\mathtt{h}}(\boldsymbol{c}_t)
\end{aligned}$$

  - $\odot$ is element-by-element multiplication
  - memory cell weight matrices ($\mathbf{W}^{\mathtt{m}}_{\mathtt{f}}, \mathbf{W}^{\mathtt{m}}_{\mathtt{i}}, \mathbf{W}^{\mathtt{m}}_{\mathtt{o}}$) diagonal
  - can allow explicit analysis of individual cell elements

Highway Connections [41]



- Gate the output of the node (example from LSTM)
  - combine with output from previous layer ($\boldsymbol{x}_t$)

$$
\begin{aligned}
\boldsymbol{i}_{\mathrm{h}} &= \boldsymbol{\sigma}(\mathbf{W}_{\mathrm{h}}^{\mathrm{f}}\boldsymbol{x}_t + \mathbf{W}_{\mathrm{h}}^{\mathrm{r}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathrm{h}}) \\
\boldsymbol{h}_t &= \boldsymbol{i}_{\mathrm{h}} \odot (\boldsymbol{i}_{\mathrm{o}} \odot \mathbf{f}^{\mathrm{h}}(\boldsymbol{c}_t)) + (\mathbf{1} - \boldsymbol{i}_{\mathrm{h}}) \odot \boldsymbol{x}_t
\end{aligned}
$$

Sequence-to-Sequence Modelling

Sequence-to-Sequence Modelling

- Sequence-to-sequence modelling central to speech/language:
  - speech synthesis:

    word sequence (discrete) $\rightarrow$ waveform (continuous)
  - speech recognition:

    waveform (continuous) $\rightarrow$ word sequence (discrete)
  - machine translation:

    word sequence (discrete) $\rightarrow$ word sequence (discrete)

- The sequence lengths on either side can differ
  - waveform sampled at 10ms/5ms frame-rate
  - word sequences (are words ...)

- Description focuses on ASR with RNNs (other models possible)

S2S: Generative Models [5, 6]

Natural Speech Technology

- Consider two sequences $L \leq T$:
  - input: $\boldsymbol{x}_{1:T} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T\}$
  - output: $\boldsymbol{y}_{1:L} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_L\}$
- Consider generative model (ASR language)

$$\begin{aligned} p(\boldsymbol{y}_{1:L}, \boldsymbol{x}_{1:T}) &= p(\boldsymbol{y}_{1:L})p(\boldsymbol{x}_{1:T}|\boldsymbol{y}_{1:L}) \\ &= p(\boldsymbol{y}_{1:L}) \sum_{\phi_{1:T} \in \boldsymbol{\Phi}_{\boldsymbol{y}_{1:L}}} p(\boldsymbol{x}_{1:T}|\phi_{1:T}) P(\phi_{1:T}|\boldsymbol{y}_{1:L}) \end{aligned}$$

  - $p(\boldsymbol{y}_{1:L})$: prior ("language") model
  - $p(\boldsymbol{x}_{1:T}|\phi_{1:T})$: (conditional) "acoustic" model
  - $P(\phi_{1:T}|\boldsymbol{y}_{1:L})$: alignment model - handles variable length

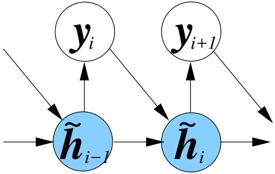
Prior Model Approximations [21, 3, 33]

- Markovian: N-gram and Feed-Forward neural network

$$p(\boldsymbol{y}_{1:L}) = \prod_{i=1}^{L} p(\boldsymbol{y}_i|\boldsymbol{y}_{1:i-1}) \approx \prod_{i=1}^{L} p(\boldsymbol{y}_i|\boldsymbol{y}_{i-1}, \ldots, \boldsymbol{y}_{i-N+1})$$

- Non-Markovian: Recurrent neural network

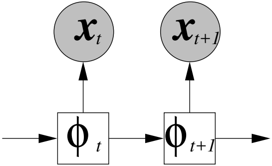
$$\begin{aligned} p(\boldsymbol{y}_{1:L}) &\approx \prod_{i=1}^{L} p(\boldsymbol{y}_i|\boldsymbol{y}_{i-1}, \tilde{\boldsymbol{h}}_{i-2}) \\ &\approx \prod_{i=1}^{L} p(\boldsymbol{y}_i|\tilde{\boldsymbol{h}}_{i-1}) \end{aligned}$$



  - depends on complete unobserved word history

Hidden Markov Models [2, 13]

- Important sequence model is the hidden Markov model (HMM)
  - an example of a dynamic Bayesian network (DBN)



- discrete latent variables
  - ϕ_t describes discrete state-space
  - conditional independence assumptions

$$P(\phi_t|\boldsymbol{\phi}_{1:t-1}, \mathbf{y}_{1:L}) = P(\phi_t|\phi_{t-1})$$
$$p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \boldsymbol{\phi}_{1:t}) = p(\mathbf{x}_t|\phi_t)$$

- The likelihood of the data is

$$p(\mathbf{x}_{1:T}|\mathbf{y}_{1:L}) = \sum_{\phi_{1:T} \in \boldsymbol{\Phi}_{\mathbf{y}_{1:L}}} \left(\prod_{t=1}^{T} p(\mathbf{x}_t|\phi_t) P(\phi_t|\phi_{t-1}) \right)$$

Acoustic Model Approximations

- Fully Markovian: HMM, simplest form of approximation

$$p(\boldsymbol{x}_{1:T}|\phi_{1:T}) \approx \prod_{t=1}^{T} p(\boldsymbol{x}_t|\phi_t)$$

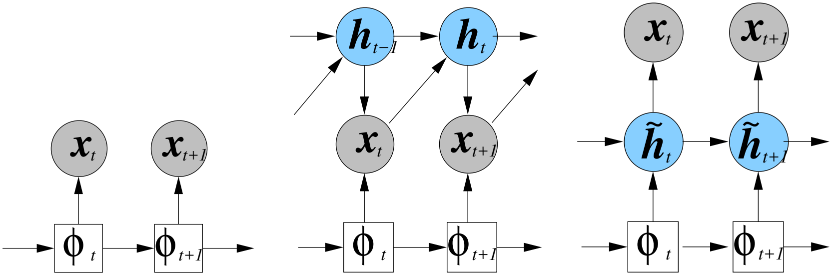
- State Markovian:

$$p(\boldsymbol{x}_{1:T}|\phi_{1:T}) \approx \prod_{t=1}^{T} p(\boldsymbol{x}_t|\phi_t, \boldsymbol{x}_{1:t-1}) \approx \prod_{t=1}^{T} p(\boldsymbol{x}_t|\phi_t, \boldsymbol{h}_{t-1})$$

- Feature Markovian:

$$p(\boldsymbol{x}_{1:T}|\phi_{1:T}) \approx \prod_{t=1}^{T} p(\boldsymbol{x}_t|\phi_{1:t}) \approx \prod_{t=1}^{T} p(\boldsymbol{x}_t|\tilde{\boldsymbol{h}}_t)$$

Markovian Approximations and Inference



Markovian (HMM)	State Markovian	Feature Markovian
$\prod_{t=1}^{T} p(\boldsymbol{x}_t \mid \phi_t)$	$\prod_{t=1}^{T} p(\boldsymbol{x}_t \mid \phi_t, \boldsymbol{h}_{t-1})$	$\prod_{t=1}^{T} p(\boldsymbol{x}_t \mid \tilde{\boldsymbol{h}}_t)$

- Inference costs significantly different:
  - state Markovian: all past history observed - deterministic
  - feature Markovian: past history unobserved - depends on path

“Likelihoods” [6]

- Deep learning can be used to estimate distributions - MDNN
  - more often trained as a discriminative model
  - need to convert to a “likelihood”
- Most common form (for RNN acoustic model):

$$p(\boldsymbol{x}_t|\phi_t, \boldsymbol{h}_{t-1}) \propto \frac{P(\phi_t|\boldsymbol{x}_t, \boldsymbol{h}_{t-1})}{P(\phi_t)}$$

  - $P(\phi_t|\boldsymbol{x}_t, \boldsymbol{h}_{t-1})$: modelled by a standard RNN
  - $P(\phi_t)$: state/phone prior probability

Why use a generative sequence-to-sequence model?

S2S: Discriminative Models [5]

- Directly compute posterior of sequence

$$p(\boldsymbol{y}_{1:L}|\boldsymbol{x}_{1:T}) = \sum_{\phi_{1:T} \in \boldsymbol{\Phi}_{\boldsymbol{y}_{1:L}}} \cancelto{1}{p(\boldsymbol{y}_{1:L}|\phi_{1:T})} P(\phi_{1:T}|\boldsymbol{x}_{1:T})$$

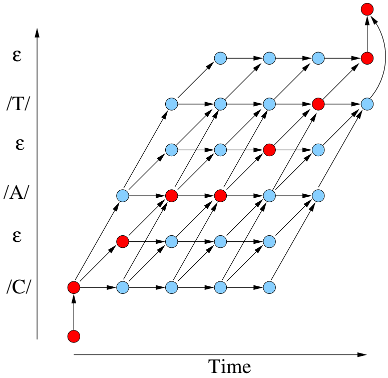
  - state Markovian RNNs used to model history/alignment

$$\begin{aligned} P(\phi_{1:T}|\boldsymbol{x}_{1:T}) &\approx \prod_{t=1}^{T} P(\phi_t|\boldsymbol{x}_{1:t}) \\ &\approx \prod_{t=1}^{T} P(\phi_t|\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) \approx \prod_{t=1}^{T} P(\phi_t|\boldsymbol{h}_t) \end{aligned}$$

- Expression does not have alignment/language models

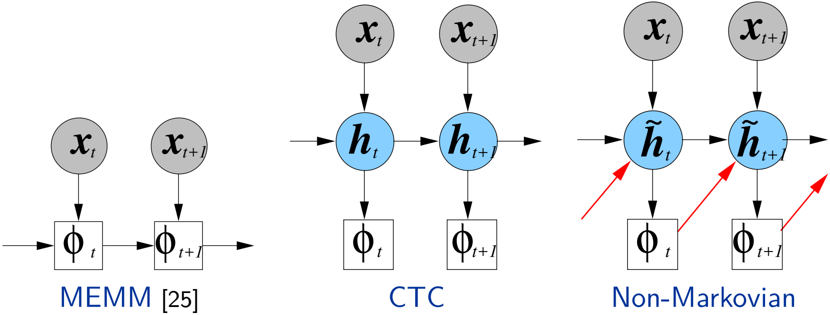
Connectionist Temporal Classification [15]

- CTC: discriminative model, no explicit alignment model
  - introduces a `blank` output symbol (ϵ)



- Consider word: `CAT`
  - Pronunciation: `/C/ /A/ /T/`
- Observe 7 frames
  - possible state transitions
  - example path:
    `/C/` ϵ `/A/` `/A/` ϵ `/T/` ϵ

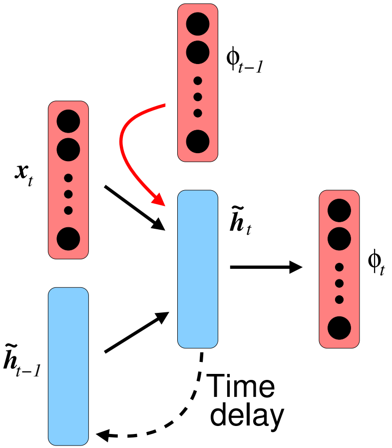
Extension to Non-Markovian



- Interesting to consider state dependencies (right)

$$P(\phi_{1:T}|\boldsymbol{x}_{1:T}) \approx \prod_{t=1}^{T} P(\phi_t|\boldsymbol{x}_{1:t}, \phi_{1:t-1}) \approx \prod_{t=1}^{T} P(\phi_t|\tilde{\boldsymbol{h}}_t)$$

Link of “Pain” - Non-Markovian



- Exact inference intractable
  - complete history dependence
  - see RNNLM ASR decoding

Discriminative Models and "Priors" [18]

Natural Speech Technology

- No language models in (this form of) discriminative model
  - in CTC the word history "captured" in state (frame) history
  - no explicit dependence on state (word) history
- Treat as a product of experts (log-linear model): for CTC

$$p(\boldsymbol{y}_{1:L}|\boldsymbol{x}_{1:T}) = \frac{1}{Z(\boldsymbol{x}_{1:T})}\exp\left(\boldsymbol{\alpha}^{\mathrm{T}}\left[\begin{array}{c}\log\left(\sum_{\phi_{1:T}\in\boldsymbol{\Phi}_{\boldsymbol{y}_{1:L}}} P(\phi_{1:T}|\boldsymbol{x}_{1:T})\right)\\ \log\left(p(\boldsymbol{y}_{1:L})\right)\end{array}\right]\right)$$

  - $\boldsymbol{\alpha}$ trainable parameter (related to LM scale)
  - $p(\boldsymbol{y}_{1:L})$ standard "prior" (language) model
- Normalisation term not required in decoding
  - $\boldsymbol{\alpha}$ often empirically tuned

S2S: Encoder-Decoder Style Models

- Directly model relationship

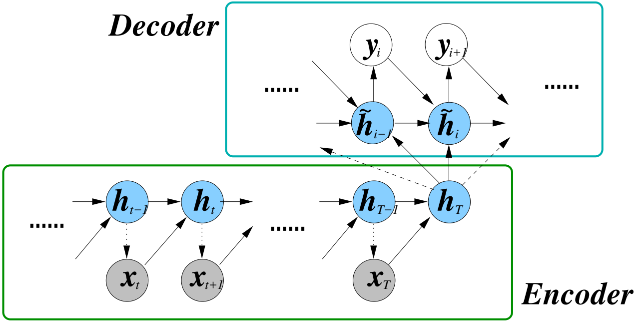
$$
\begin{aligned}
p(\boldsymbol{y}_{1:L}|\boldsymbol{x}_{1:T}) &= \prod_{i=1}^{L} p(\boldsymbol{y}_i|\boldsymbol{y}_{1:i-1}, \boldsymbol{x}_{1:T}) \\
&\approx \prod_{i=1}^{L} p(\boldsymbol{y}_i|\boldsymbol{y}_{i-1}, \tilde{\boldsymbol{h}}_{i-2}, \boldsymbol{c})
\end{aligned}
$$

  - looks like an RNN LM with additional dependence on $\boldsymbol{c}$

$$\boldsymbol{c} = \phi(\boldsymbol{x}_{1:T})$$

  - $\boldsymbol{c}$ is a fixed length vector - like a sequence kernel

RNN Encoder-Decoder Model [14, 32]

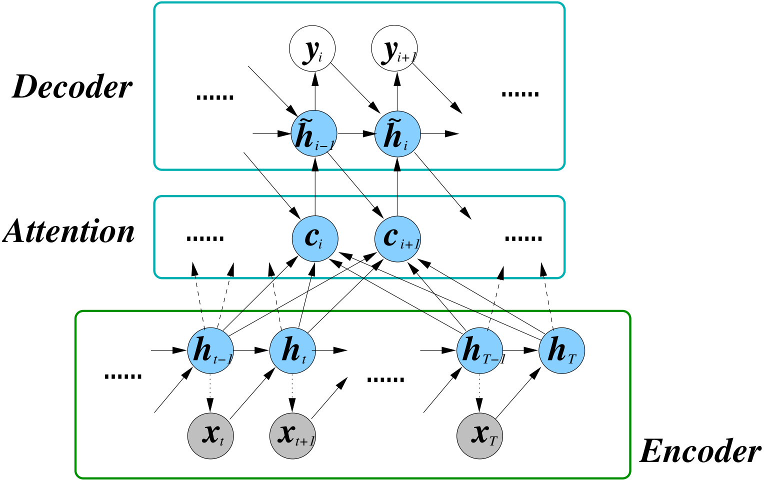


- Simplest form is to use hidden unit from acoustic RNN/LSTM

$$\boldsymbol{c} = \phi(\boldsymbol{x}_{1:T}) = \boldsymbol{h}_T$$

  - dependence on context is global via $\boldsymbol{c}$ - possibly limiting

Attention-Based Models [9, 7, 32]



Attention-Based Models

- Introduce attention layer to system
  - introduce dependence on locality i

$$p(\boldsymbol{y}_{1:L}|\boldsymbol{x}_{1:T}) \approx \prod_{i=1}^{L} p(\boldsymbol{y}_i|\boldsymbol{y}_{i-1}, \tilde{\boldsymbol{h}}_{i-2}, \boldsymbol{c}_i) \approx \prod_{i=1}^{L} p(\boldsymbol{y}_i|\tilde{\boldsymbol{h}}_{i-1})$$

$$\boldsymbol{c}_i = \sum_{\tau=1}^{T} \alpha_{i\tau}\boldsymbol{h}_\tau; \quad \alpha_{i\tau} = \frac{\exp(e_{i\tau})}{\sum_{k=1}^{T}\exp(e_{ik})}, \quad e_{i\tau} = f^{\mathrm{e}}\left(\tilde{\boldsymbol{h}}_{i-2}, \boldsymbol{h}_\tau\right)$$

  - $e_{i\tau}$ how well position $i-1$ in input matches position τ in output
  - $\boldsymbol{h}_\tau$ is representation (RNN) for the input at position τ
- Attention can "wander" with large input size (T)
  - use a pyramidal network to reduce frame-rate for attention

S2S: Structured Discriminative Models [12, 31, 50]

- General form of structured discriminative model

$$p(\boldsymbol{y}_{1:L}|\boldsymbol{x}_{1:T}) = \frac{1}{Z(\boldsymbol{x}_{1:T})} \sum_{\phi_{1:T} \in \boldsymbol{\Phi}_{\boldsymbol{y}_{1:L}}} \exp\left(\boldsymbol{\alpha}^{\mathsf{T}} \boldsymbol{f}(\boldsymbol{x}_{1:T}, \phi_{1:T}, \boldsymbol{y}_{1:L})\right)$$

  - $\boldsymbol{f}(\boldsymbol{x}_{1:T}, \phi_{1:T}, \boldsymbol{y}_{1:L})$ extracts features from observations/states/words
  - need to map variable length sequence to a fixed length (again)
  - latent variables, state sequence $\phi_{1:T}$, "aid" attention

- Integrate with deep learning to model segment features:
  - RNN to map segments to a fixed length vector
  - segment posterior outputs from multiple systems (joint decoding)

Natural Speech Technology

Speech Processing Applications

LM: Neural Network Language Models [3, 33]

Natural Speech Technology

- Neural networks extensively used for language modelling
  - recurrent neural networks - complete word history

$$P(\boldsymbol{\omega}_{1:L}) = \prod_{i=1}^{L} P(\omega_i|\boldsymbol{\omega}_{1:i-1}) \approx \prod_{i=1}^{L} P(\omega_i|\omega_{i-1}, \tilde{\boldsymbol{h}}_{i-2}) \approx \prod_{i=1}^{L} P(\omega_i|\tilde{\boldsymbol{h}}_{i-1})$$

- 1-of-K ("one-hot") coding for ith word, ω_i, $\boldsymbol{y}_i$
  - additional out-of-shortlist symbol may be added
  - softmax activation function on output layer

- Issues that need to be addressed
  1. training: how to efficiently train on billions of words?
  2. decoding: how to handle dependence on complete history?

LM: Cross-Entropy Training Criteria

- Standard training criterion for word sequence $\boldsymbol{\omega}_{1:L} = \omega_1, \ldots, \omega_L$

$$\mathcal{F}_{\text{ce}} = -\frac{1}{L}\sum_{i=1}^{L} \log\left(P(\omega_i|\tilde{\boldsymbol{h}}_{i-1})\right)$$

  - GPU training makes this reasonable BUT
- Compute cost for softmax normalisation term $Z(\tilde{\boldsymbol{h}}_{i-1})$

$$P(\omega_i|\tilde{\boldsymbol{h}}_{i-1}) = \frac{1}{Z(\tilde{\boldsymbol{h}}_{i-1})}\exp\left(\boldsymbol{w}_{f(\omega_i)}^{\mathsf{T}}\tilde{\boldsymbol{h}}_{i-1}\right)$$

  - required as unobserved sequence (contrast acoustic model)

LM: Alternative Training Criteria [8]

- Variance Regularisation: eliminate normalisation from decoding

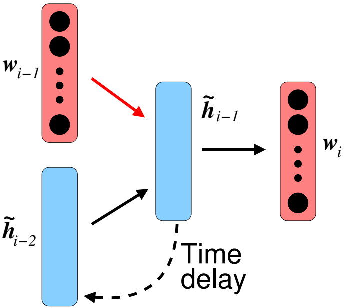
$$\mathcal{F}_{\texttt{vr}} = \mathcal{F}_{\texttt{ce}} + \frac{\gamma}{2}\frac{1}{L}\sum_{i=1}^{L}\left(\log(Z(\tilde{\boldsymbol{h}}_{i-1})) - \overline{\log(Z)}\right)^2$$

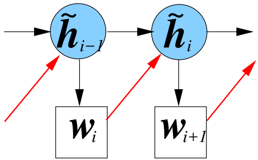
  - $\overline{\log(Z)}$ average (log) history normalisation
  - all normalisation terms tend to be the same
- Noise Contrastive Estimation: efficient decoding and training

$$\mathcal{F}_{\texttt{nce}} = -\frac{1}{L}\sum_{i=1}^{L}\left(\log(P(y_i = \texttt{T}|\omega_i, \tilde{\boldsymbol{h}}_{i-1}) + \sum_{j=1}^{k}\log(P(y_i = \texttt{F}|\hat{\omega}_{ij}, \tilde{\boldsymbol{h}}_{i-1})\right)$$

  - $\hat{\omega}_{ij}$ are competing samples for ω_i - often sample from uni-gram LM

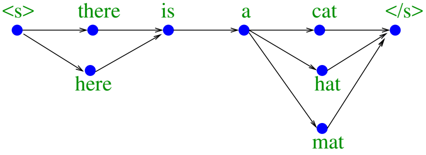
LM: ASR Decoding with RNNLMs



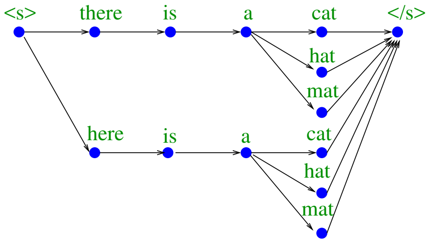


- ASR decoding with RNNLM has a link of "pain"
    - history vector depends on "unobserved" word sequence

LM: ASR Decoding with RNNLMs



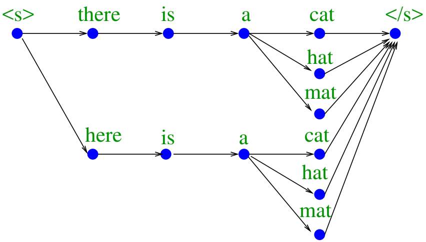
Lattice



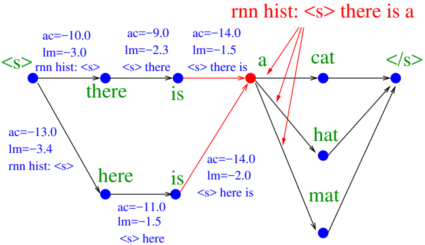
Prefix Tree

- Consider word-lattice on the left
  - expands to prefix tree (right) if complete history taken into account
  - significant increase in number of paths

LM: N-Gram History Approximation [29]



Prefix Tree



N-Gram Approximation

- Use exact RNN LM value but
  - merge paths based on N-gram history
  - can also use history vector distance merging

ASR: Sequence Training [24]

- Cross-Entropy using fixed alignment standard criterion (RNN)

$$\mathcal{F}_{\texttt{ce}} = -\sum_{t=1}^{T} \log\left(P(\hat{\phi}_t | \boldsymbol{x}_t, \boldsymbol{h}_{t-1})\right)$$

  - criterion based on frame-by-frame classification
- Sequence training integrates sequence modelling into training

$$\mathcal{F}_{\texttt{mbr}} = \sum_{r=1}^{R} \sum_{\tilde{\omega}} P(\tilde{\omega} | \boldsymbol{x}_{1:T}^{(r)}) \mathcal{L}(\tilde{\omega}, \omega^{(r)})$$

  - MBR described (various loss functions) - also CML used
  - may be applied to generative and discriminative models

ASR: Sequence Training and CTC [36]

- Sequence-training is discriminative, so is CTC ...
  - let's ignote the `blank` symbol
  - consider MMI as the training criterion

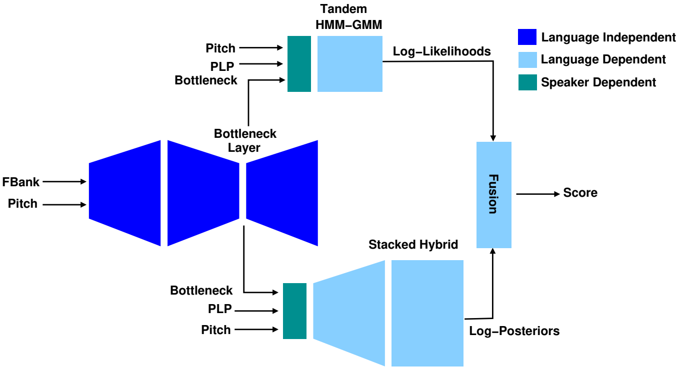
$$\mathcal{F}_{\texttt{mmi}} = \sum_{r=1}^{R} \log\left(P(\boldsymbol{\omega}^{(r)}|\boldsymbol{x}_{1:T}^{(r)})\right)$$

  - lattice-free training uses some of the CTC-style approaches

- CTC has local (every frame) normalistion
  - discriminatively-trained generative models use global normalisation
- CTC has no "language-model"
  - use phone level language model $P(\texttt{ph}_i|\texttt{ph}_j)$ (a 4-gram used)

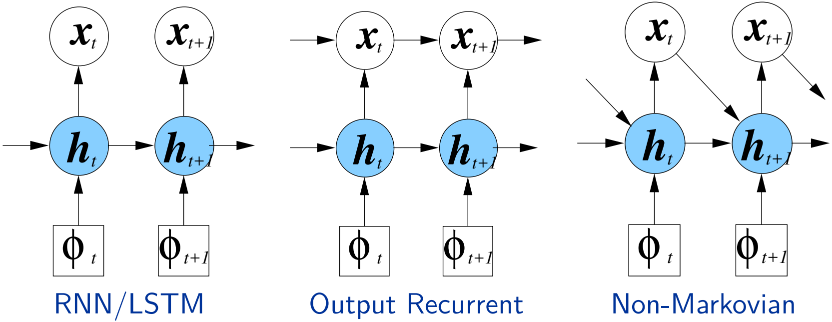
ASR: Joint Decoding (Stacked Hybrid) [50]

Natural Speech Technology



- DNN acoustic models and Tandem systems
  - uses bottleneck features and stacking
  - fusion: based on log-linear models - structured SVM

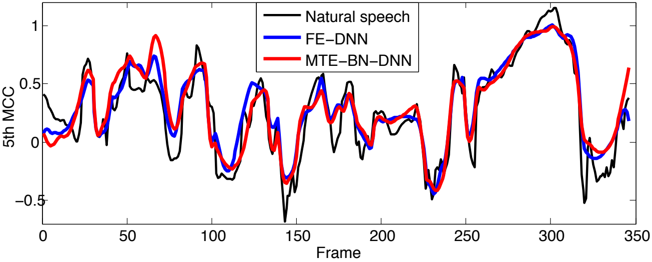
TTS: Deep Learning for Synthesis [53, 28]



- For statistical speech synthesis model

$$p(\boldsymbol{x}_{1:T}|\boldsymbol{y}_{1:L}) = \sum_{\phi_{1:T} \in \boldsymbol{\Phi}_{\boldsymbol{y}_{1:L}}} p(\boldsymbol{x}_{1:T}|\phi_{1:T}) P(\phi_{1:T}|\boldsymbol{y}_{1:L}) \approx p(\boldsymbol{x}_{1:T}|\hat{\phi}_{1:T})$$

TTS: Bottleneck Features [49]



- Can also include bottleneck features (similar to ASR)
  - FE-DNN: standard feed-forward NN
  - MTE-BN-DNN: feed-forward NN with MTE and BN features
- Subjectively better as well

TTS: Minimum Trajectory Error [47]

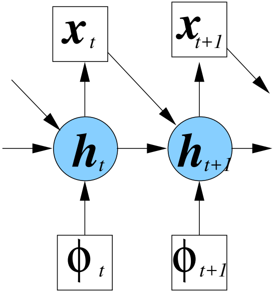
- Smoothing is an important issue for generating trajectories in TTS
  - predict experts for static and dynamics parameters (MLPG)
  - use recursion on the output layer
- When using experts can minimise trajectory error (MTE)

$$\begin{aligned}
\mathcal{F}_{\texttt{mge}} &= \sum_{r=1}^{R} (\hat{\boldsymbol{x}}_{1:T}^{(r)} - \boldsymbol{x}_{1:T}^{(r)})^{\texttt{T}} (\hat{\boldsymbol{x}}_{1:T}^{(r)} - \boldsymbol{x}_{1:T}^{(r)}) \\
&= \sum_{r=1}^{R} (\boldsymbol{R}\hat{\boldsymbol{o}}_{1:T}^{(r)} - \boldsymbol{x}_{1:T}^{(r)})^{\texttt{T}} (\boldsymbol{R}\hat{\boldsymbol{o}}_{1:T}^{(r)} - \boldsymbol{x}_{1:T}^{(r)})
\end{aligned}$$

  - $\hat{\boldsymbol{o}}_{1:T}^{(r)}$ is the sequence of static/delta (means)
  - $\boldsymbol{R}$ is the mapping matrix from static/deltas to trajectory

TTS: Conditional Language Modelling [44]

Natural Speech Technology

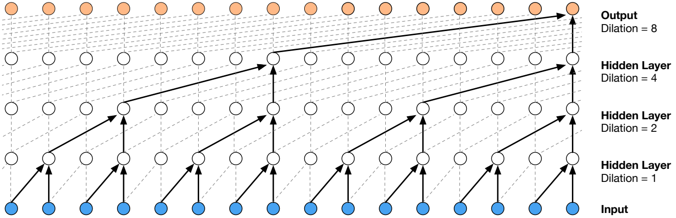


- Make the system non-Markovian
- 1-of-K coding for samples
  - sample-level synthesis
  - 8-bit = 256 output
  - softmax output activation
- Recurrent units (shown):
  - 240ms=3840 samples
  - insufficient history memory

- Replace recurrent units by a sparse Markovian history
  - extensive use of dilation to limit model parameters
- Search not an issue - simply sampling!

TTS: CNN and RNN History Modelling [51, 44]

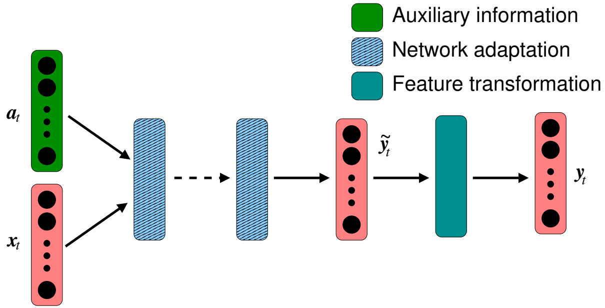
- Markovian history modelling limimted by parameter growth
  - network parameters grows linearly with the length of history
- Non-Markovan (recurrent) approaches address parameter issue
  - but hard to train, and long-term representation (often) poor



- CNN with dilation an alternative balance
  - exponential expansion of history, limited parameeter growth

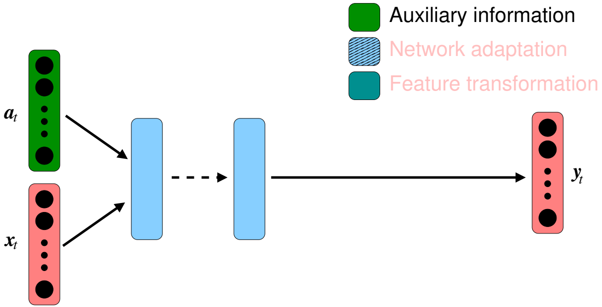
Neural Network Adaptation

Neural Network Adaptation



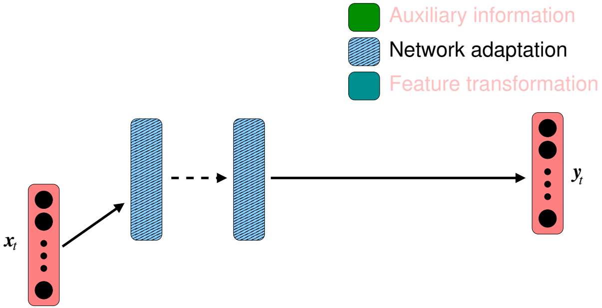
- Similar approaches used for TTS/ASR/LMs - broad classes
  - auxiliary information: i-vectors, gender information, emotion]
  - network adaptation: weight adaptation, activation function adaptation
  - feature transformation: linear transformation of the features
- Possible to combine multiple approaches together

Auxiliary Information [34, 39, 22, 23, 8]



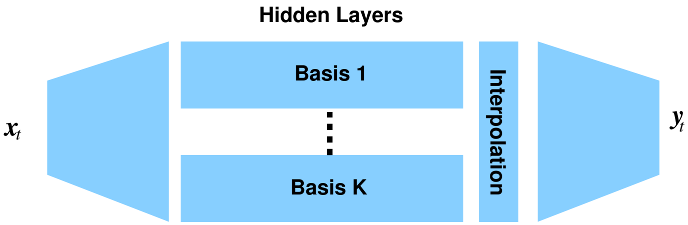
- Include information about speaker/topic/environment to network
  - often vector representation used - iVectors, LDA topic spaces
  - possible to adapt to speaker without hypothesis (e.g. iVectors)
- Applied to language modelling, speaker and noise adaptation

Network Parameter Adaptation



- Seen a range of network adaptation approaches for ASR
  - use "well-trained" network parameters as a prior
  - updates all the parameters, weights, of the system
- Is it possible to reduce number of parameters to be adapted?

Structuring Networks for Adaptation [45]



- Structure network as a series of bases
  - interpolation layer is speaker dependent

$$\mathbf{h}^{(ls)} = \sum_{k=1}^{K} \lambda_k^{(s)} \mathbf{h}_k^{(l)}$$

  - few parameters per speaker - very rapid adaptation
  - interpolation estimation convex optimisation problem

Activation Function Adaptation [42, 54, 43]

Natural Speech Technology

- Consider a layer of a network with 1000×1000 connections
  - weights: 1,000,000 parameters to adjust
  - activation functions: 2,000 functions (output and input)
- Take the example of a sigmoid activation function

$$\phi_i(\alpha_{\mathtt{i}}^{(s)}, \alpha_{\mathtt{o}}^{(s)}, \alpha_{\mathtt{b}}^{(s)}) = \frac{\alpha_{\mathtt{o}}^{(s)}}{1 + \exp\left(\alpha_{\mathtt{i}}^{(s)} \mathbf{w}_i^{\mathsf{T}} \boldsymbol{x}_t + \alpha_{\mathtt{b}}^{(s)}\right)}$$

  - $\alpha_{\mathtt{i}}^{(s)}$: scaling of the input
  - $\alpha_{\mathtt{o}}^{(s)}$: scaling of the output
  - $\alpha_{\mathtt{b}}^{(s)}$: offset on the activation
  - train these (or subset) parameters to be speaker specific
- Exact form of parameter adaptation depends on activation function

Factored Weight Adaptation [30]

- Consider a speaker-specific linear transform of the weight matrix

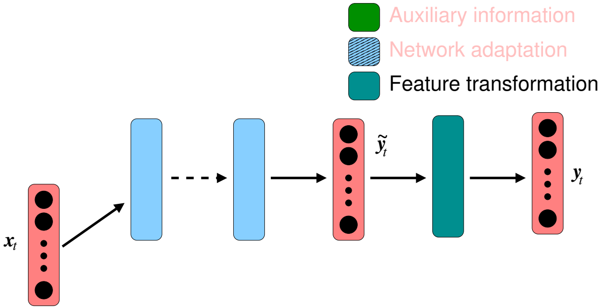
$$\mathbf{W}^{(s)} = \mathbf{A}^{(s)}\mathbf{W}$$

  - act on a low-dimensional compression layer
- Compact transform central to aspects like covariance modelling:

$$\mathbf{W}^{(s)} = \mathbf{W} + \sum_{i=1}^{P} \lambda_i^{(s)} \mathbf{A}^{(i)}\mathbf{W}$$

  - number of parameters is P - independent of weight matrix size
  - $\mathbf{A}^{(i)}$ low-rank, limits number of parameters
  - can make a function of auxiliary information (e.g. iVectors)

Feature Transformation (TTS) [11, 48]



- Train the network to predict experts for normalised features, $\tilde{\boldsymbol{y}}_t$,
  - transform normalised experts to target speaker experts
- ASR constrained to use global transform
  - TTS can make use of regression class trees (CSMAPLR)

NST and Deep Learning

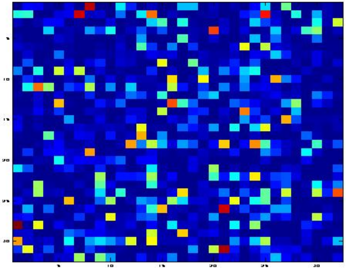
Natural Speech Technology

- Many other applications of deep learning under NST
- Speech Recognition
  - audio data segmentation
  - beamforming and channel combination
  - network initialisation and regularisation
  - acoustic feature dependency modelling
  - paraphrastic language models
- Speech Synthesis
  - post-processing
  - influence/limitations of LSTM parameters on synthesis
  - robust duration modelling
  - unit selection

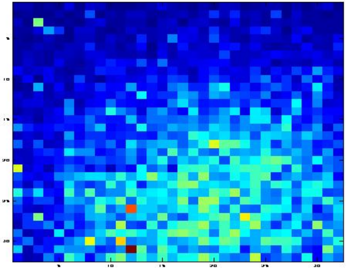
Is Deep Learning the Solution?

- Most research still uses a two-stage approach to training:
  1. feature extraction: convert waveform to parametric form
  2. modelling: given parameters train model
- Limitations in the feature extraction stage cannot be overcome ...
  - integrate feature extraction into process
  - attempt to directly model/synthesise waveform (WaveNet)
- Both are interesting, active, areas of research
  - links with "integrated end-to-end" systems: waveform-in words-out
  - feasible as quantity of data increases
- BUT
  - networks are difficult to optimise - tuning required
  - hard to interpret networks to get insights
  - sometimes difficult to learn from previous tasks ...

Network Interpretation [46]



Standard /ay/



Stimulated /ay/

- Deep learning usually highly distributed - hard to interpret
  - awkward to adapt/understand/regularise
  - modify training - add stimulation regularisation (improves ASR!)

Thank-you!

[1] O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," in *2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP)*. IEEE, 2012, pp. 4277–4280.

[2] L. Baum and J. Eagon, "An Inequality with Applications to Statistical Estimation for Probabilistic Functions of Markov Processes and to a Model for Ecology," *Bull Amer Math Soc*, vol. 73, pp. 360–363, 1967.

[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[4] C. Bishop, "Mixture density networks," in *Tech. Rep. NCRG/94/004, Neural Computing Research Group, Aston University*, 1994.

[5] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer Verlag, 2006.

[6] H. Bourlard and N. Morgan, "Connectionist speech recognition: A hybrid approach," 1994.

[7] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *CoRR*, vol. abs/1508.01211, 2015. [Online]. Available: http://arxiv.org/abs/1508.01211

[8] X. Chen, T. Tan, X. Liu, P. Lanchantin, M. Wan, M. J. Gales, and P. C. Woodland, "Recurrent neural network language model adaptation for multi-genre broadcast speech recognition," in *Proceedings of InterSpeech*, 2015.

[9] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *CoRR*, vol. abs/1506.07503, 2015. [Online]. Available: http://arxiv.org/abs/1506.07503

[10] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," *CoRR*, vol. abs/1506.02216, 2015. [Online]. Available: http://arxiv.org/abs/1506.02216

[11] S. Deena, M. Hasan, M. Doulaty, O. Saz, and T. Hain, "Combining feature and model-based adaptation of RNNLMs for multi-genre broadcast speech recognition," in *Proceedings of the 17th Annual Conference of the International Speech Communication Association (Interspeech)*, 2016.

[12] M. J. F. Gales, S. Watanabe, and E. Fosler-Lussier, "Structured discriminative models for speech recognition: An overview," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 70–81, 2012.

[13] M. Gales and S. Young, "The application of hidden Markov models in speech recognition," *Foundations and Trends in Signal Processing*, vol. 1, no. 3, 2007.

[14] A. Graves, "Sequence transduction with recurrent neural networks," *CoRR*, vol. abs/1211.3711, 2012. [Online]. Available: http://arxiv.org/abs/1211.3711

[15] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[16] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[18] G. E. Hinton, "Products of experts," in *Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN 99)*, 1999, pp. 1–6.

[19] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[21] F. Jelinek, *Statistical methods for speech recognition*, ser. Language, speech, and communication. Cambridge (Mass.), London: MIT Press, 1997.

[22] P. Karanasou, M. Gales, and P. Woodland, "I-vector estimation using informative priors for adaptation of deep neural networks," in *Proc. of Interspeech*, 2015.

[23] P. Karanasou, Y. Wang, M. Gales, and P. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *Proceedings of Interspeech'14*, 2014.

[24] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 3761–3764.

[25] H.-K. Kuo and Y. Gao, "Maximum entropy direct models for speech recognition," *IEEE Transactions Audio Speech and Language Processing*, 2006.

[26] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[27] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[28] Z.-H. Ling, S.-Y. Kang, H. Zen, A. Senior, M. Schuster, X.-J. Qian, H. M. Meng, and L. Deng, "Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 35–52, 2015.

[29] X. Liu, Y. Wang, X. Chen, M. Gales, and P. Woodland, "Efficient lattice rescoring using recurrent neural network language models," in *IEEE ICASSP2014*, IEEE ICASSP2014. IEEE ICASSP2014, 04/05/2014 2014.

[30] Y. Liu, P. Karanasou, and T. Hain, "An investigation into speaker informed dnn front-end for LVCSR," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, April 2015.

[31] L. Lu, L. Kong, C. Dyer, N. A. Smith, and S. Renals, "Segmental recurrent neural networks for end-to-end speech recognition," in *Proc. INTERSPEECH*, 2016.

[32] L. Lu, X. Zhang, K. Cho, and S. Renals, "A study of the recurrent neural network encoder-decoder for large vocabulary speech recognition," in *Proc. INTERSPEECH*, 2015.

[33] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model." in *Interspeech*, vol. 2, 2010, p. 3.

[34] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model." in *SLT*, 2012, pp. 234–239.

[35] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.

[36] D. Povey, V. Peddinti, D. Galvez, P. Ghahrmani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," *Interspeech*, 2016.

[37] T. Robinson and F. Fallside, "A recurrent error propagation network speech recognition system," *Computer Speech & Language*, vol. 5, no. 3, pp. 259–274, 1991.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1," D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, Eds. Cambridge, MA, USA: MIT Press, 1986, ch. Learning Internal Representations by Error Propagation, pp. 318–362.

[39] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors." in *ASRU*, 2013, pp. 55–59.

[40] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[41] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *CoRR*, vol. abs/1505.00387, 2015. [Online]. Available: http://arxiv.org/abs/1505.00387

[42] P. Swietojanski and S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Proc. IEEE Workshop on Spoken Language Technology*, December 2014.

[43] ——, "Differentiable pooling for unsupervised acoustic model adaptation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. PP, no. 99, pp. 1–1, 2016.

[44] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw auaio," *CoRR*, 2016. [Online]. Available: https://arxiv.org/pdf/1609.03499.pdf

[45] C. Wu and M. Gales, "Multi-basis adaptive neural network for rapid adaptation in speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, IEEE. IEEE, 2015.

[46] C. Wu, P. Karanasou, M. Gales, and K. C. Sim, "Stimulated deep neural network for speech recognition," in *Proceedings interspeech*, 2016.

[47] Z. Wu and S. King, "Minimum trajectory error training for deep neural networks, combined with stacked bottleneck features," in *Proc. Interspeech*, 2015, pp. 309–313.

[48] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, "A study of speaker adaptation for dnn-based speech synthesis," in *Proceedings interspeech*, 2015.

[49] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015.

[50] J. Yang, C. Zhang, A. Ragni, M. Gales, and P. Woodland, "System combiantion with log-linear models," in *Proc. ICASSP'16*, Shanghai, China, 2016.

[51] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *CoRR*, vol. abs/1511.07122, 2015. [Online]. Available: http://arxiv.org/abs/1511.07122

[52] H. Zen and A. Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 3844–3848.

[53] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2013, pp. 7962–7966.

[54] C. Zhang and P. Woodland, "DNN speaker adaptation using parameterised sigmoid and ReLU hidden activation functions," in *Proc. ICASSP'16*, 2016.