Robustness (cont.); End-to-end systems

Steve Renals

Automatic Speech Recognition – ASR Lecture 18
27 March 2017

Robust Speech Recognition

Additive Noise

- Multiple acoustic sources are the norm rather than the exception
- From the point of view of trying to recognize a single stream of speech, this is additive noise
- Stationary noise: frequency spectrum does not change over time (e.g. air conditioning, car noise at constant speed)
- Non-stationary noise: time-dependent frequency spectrum (e.g. breaking glass, workshop noise, music, speech)
- Measure the noise level as SNR (signal-to-noise ratio), measured in dB
  - 30dB SNR sounds noise free
  - 0dB SNR has equal signal and noise energy

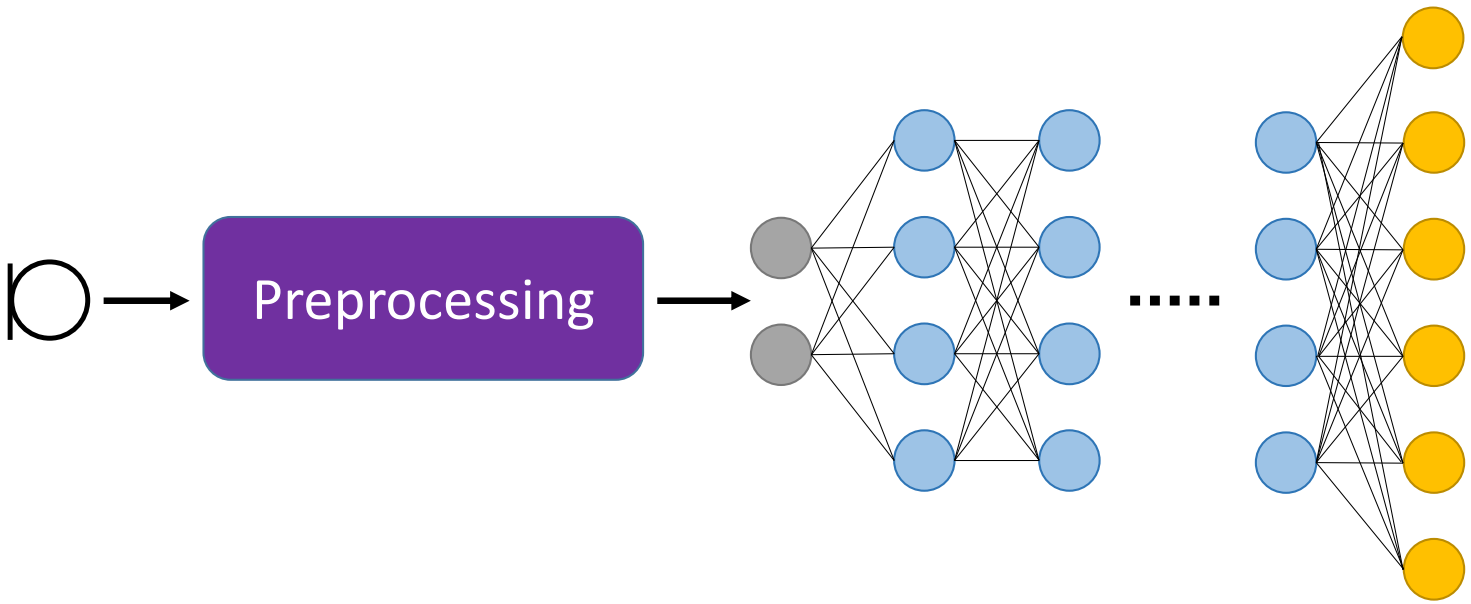
Approaches to Robust Speech Recognition

- **Feature normalisation:** Transform the features to reduce mismatch between training and test (e.g. CMN/CVN)
- **Feature compensation:** Estimate the noise spectrum and subtract it from the observed spectra e.g. sprectral subtraction
- **Multi-condition training** — train with speech data in a variety of noise conditions. It is possible to artificially mix recorded noise with clean speech at any desired SNR to create a multi-style training set
- **Model adaptation** — use an adaptation technique such as MLLR to adapt to the acoustic environment

Current approaches to robust speech recognition

Decoupled preprocessing: Acoustic processing independent of downstream activity

- Pro: simple
- Con: removes variability
- Example: beamforming for multi-microphone distant speech recognition

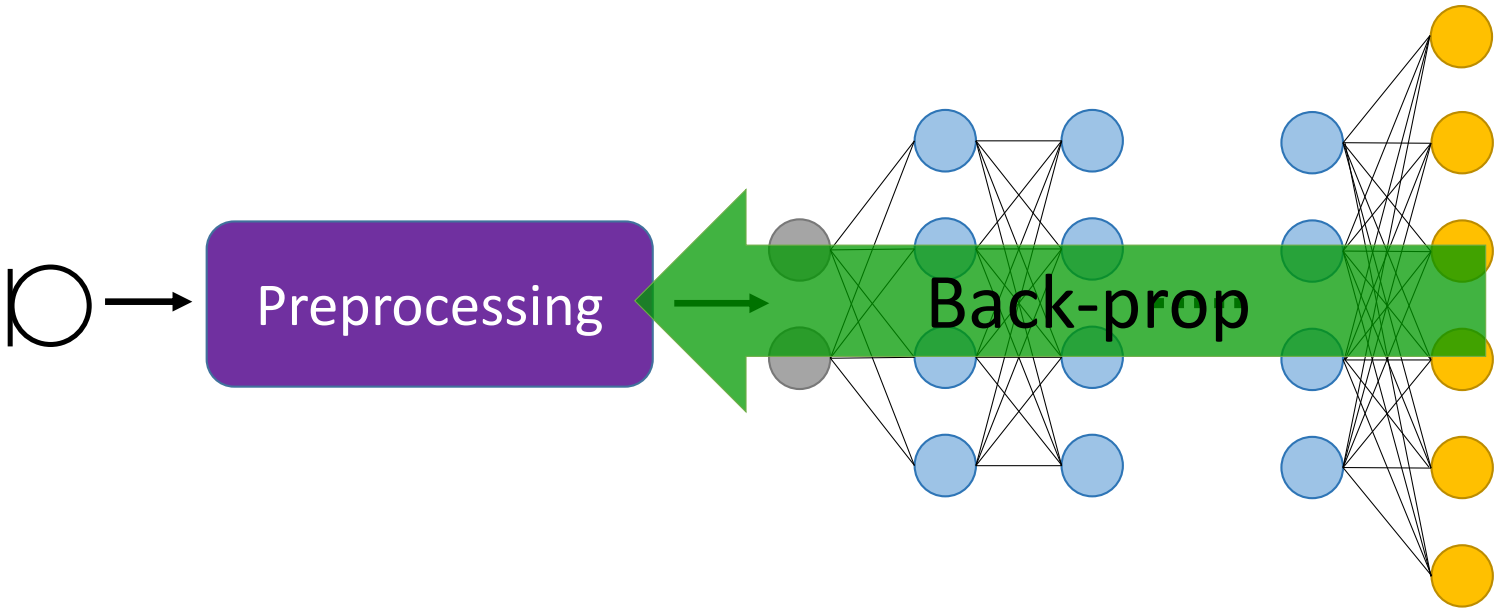


Slide from Mike Seltzer

Current approaches to robust speech recognition

Integrated processing: Treat acoustic processing as initial layers of the network – optimise parameters with back propagation

- Pro: should be "optimal" for the model
- Con: computationally expensive,
- Example: direct waveform systems

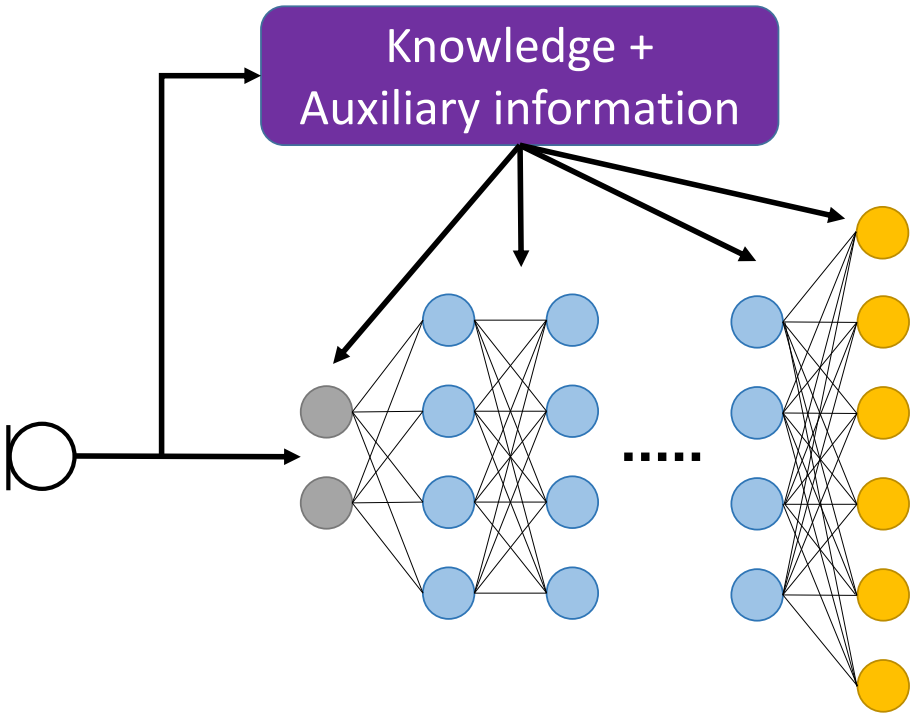


Slide from Mike Seltzer

Current approaches to robust speech recognition

Augmented information: Add additional side information to the network (additional nodes, different objective function, ...)

- Pros: preserves variability, adds knowledge, maintains representation
- Con: not a physical model
- Example: noise-aware training, factorised "noise codes" (iVectors)



Slide from Mike Seltzer

End-to-End Modelling

Limitations of HMMs

- Sequence trained HMM/NN systems have limitations
  - Markov assumption – current state depends on only the previous state
  - Conditional independence assumptions – dependence on previous acoustic observations encapsulated in the current state
- RNNs are powerful sequence models
  - recurrent hidden state much richer history representation than HMM state
  - can learn representations
  - can directly model dependences through time
- But HMM/RNN systems only use RNNs to model time within a phone / HMM state...

"End-to-end" ("HMM-Free") RNN speech recognition

- **Can RNNs replace the HMM sequence model?**
- Yes – active research topic. On approach is to use an **RNN encoder-decoder** model
- The **encoder** maps the the input sequence vector into a sequence of RNN hidden states
- The **decoder** maps the RNN hidden states into an output sequence
- Input and output sequences may be different lengths
  - Input sequence of frames
  - Output sequence of phones or letters or words!
- Mapping to directly to words results in a joint acoustic and language model

RNN Encoder-Decoder

- The overall task is to compute the probability of an output sequence given an input sequence,
  $P(\mathbf{y}_1, \ldots, \mathbf{y}_O | \mathbf{x}_1, \ldots, \mathbf{x}_T) = P(\mathbf{y}_1^O | \mathbf{x}_1^T)$
- **Encoder:** compute a *context* $\mathbf{c}_o$ for each output $\mathbf{y}_o$
- **Decoder:** compute

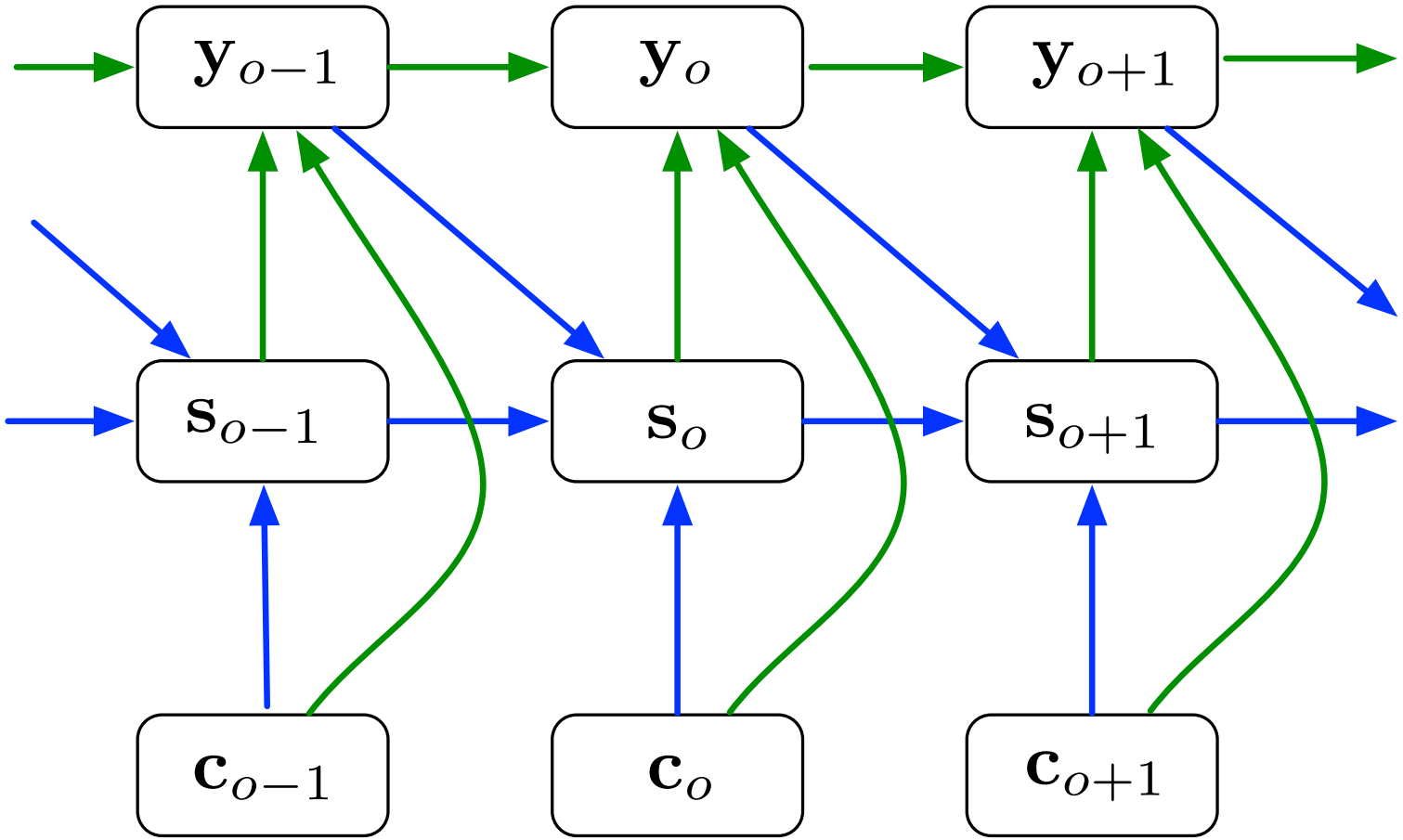
$$P(\mathbf{y}_1^O | \mathbf{x}_1^T) = \prod_o \underbrace{P(\mathbf{y}_o | \mathbf{y}_1^{o-1}, \mathbf{c}_o)}_{\text{RNN}}$$

$$P(\mathbf{y}_o | \mathbf{y}_1^{o-1}, \mathbf{c}_o) = \text{softmax}(\mathbf{y}_{o-1}, \mathbf{s}_o, \mathbf{c}_o)$$

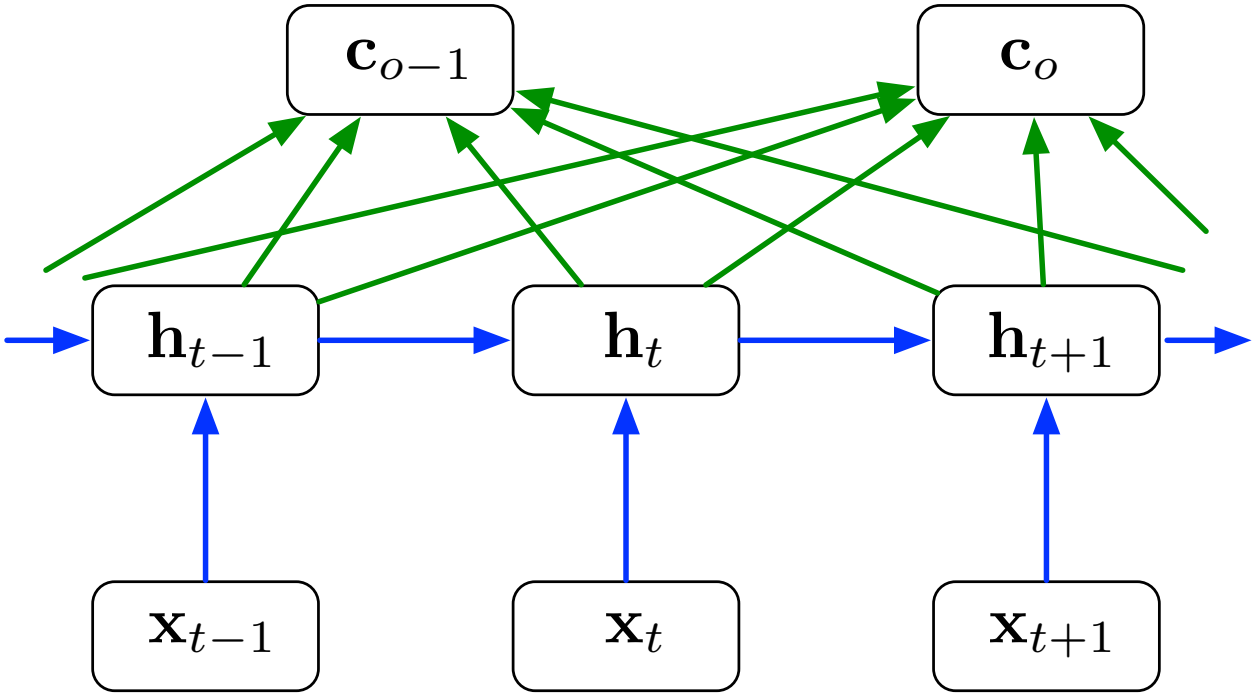
$$\mathbf{s}_o = f(\mathbf{y}_{o-1}, \mathbf{s}_{o-1}, \mathbf{c}_o)$$

  - $\mathbf{y}_{o-1}$ is the previous output
  - $\mathbf{s}_o$ is the decoder state (recurrent hidden layer)
  - $\mathbf{c}_o$ is the encoder context

RNN decoder


$\mathbf{y}_{o-1}$
$\mathbf{y}_o$
$\mathbf{y}_{o+1}$
$\mathbf{s}_{o-1}$
$\mathbf{s}_o$
$\mathbf{s}_{o+1}$
$\mathbf{c}_{o-1}$
$\mathbf{c}_o$
$\mathbf{c}_{o+1}$

RNN encoder



$$\mathbf{c}_o = \sum_t \alpha_{ot} \mathbf{h_t}$$

$$\alpha_{ot} = \underbrace{\text{softmax}(g(\mathbf{s}_{o-1}, \mathbf{h}_t))}_{\text{NN}}$$

RNN encoder-decoder

- Train all the parameters to maximise $\log P(\mathbf{y}_1^O|\mathbf{x}_1^T)$ using backprop through time
- The encoder is a bidirectional RNN
- Training/testing on Switchboard, directly mapping MFCCs to words (no pronunciation model, no language model) gives 49% WER
- Improved training scheme, FBANK features gives 37% WER
- Potential improvements
  - multiple recurrent layers in the encoder
  - incorporating a language model in the decoder
  - using character-based output sequence

(L Lu et al (2015), "A Study of the Recurrent Neural Network Encoder-Decoder for Large Vocabulary Speech Recognition", Interspeech-2015, http://homepages.inf.ed.ac.uk/llu/pdf/liang_is15a.pdf)