Encoder-decoder models 2: attention-based models

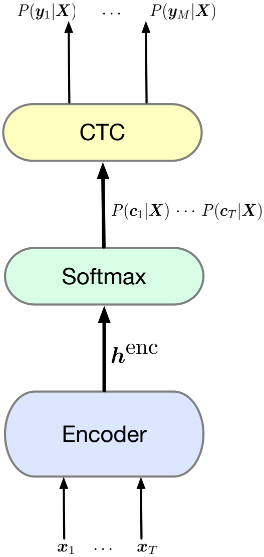
Peter Bell

Automatic Speech Recognition – ASR Lecture 14
5 March 2026

Recap: CTC

View CTC as having three components:

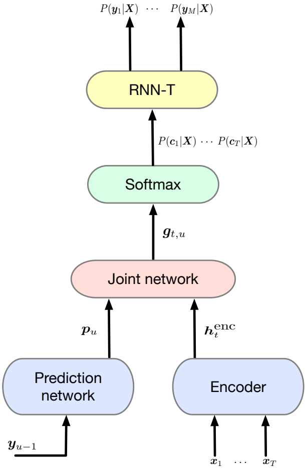
- **Encoder**: Deep (bidirectional) LSTM recurrent network which maps acoustic features $X = x_1, \ldots, x_T$ to a sequence of hidden vectors $h^{\text{enc}} = h_1^{\text{enc}}, \ldots, h_T^{\text{enc}}$.
- **Softmax**: Computes the label probabilities $P(c_1|X), \ldots, P(c_T|X)$
- **CTC**: Computes the subword sequence $P(y_1|X), \ldots, P(y_M|X)$



$P(\boldsymbol{y}_1|\boldsymbol{X})$ … $P(\boldsymbol{y}_M|\boldsymbol{X})$
CTC
$P(\boldsymbol{c}_1|\boldsymbol{X}) \cdots P(\boldsymbol{c}_T|\boldsymbol{X})$
Softmax
$\boldsymbol{h}^{\text{enc}}$
Encoder
$\boldsymbol{x}_1$ … $\boldsymbol{x}_T$

Recap: RNN-T

- **Encoder:** Acoustic model network mapping acoustic features $X = x_1, \ldots, x_T$ to hidden vectors $h^{\text{enc}} = h_1^{\text{enc}}, \ldots, h_T^{\text{enc}}$.
- **Prediction network**: Recurrent network which takes the previous output subword label y_{u-1} as input and predicts the next subword label p_u – acts as a language model (over subwords)
- **Joint network**: Computes a joint hidden vector $g_{t,u}$ by a applying a shallow feed-forward net to h^{enc} and p_u
- Followed by **softmax** and **CTC** components as before

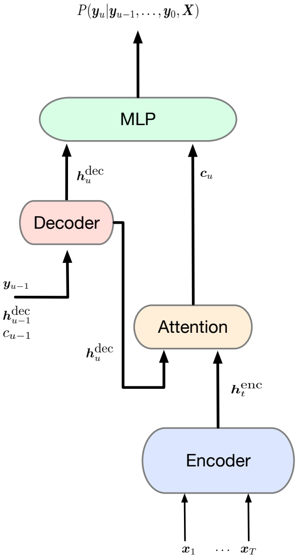

$P(\boldsymbol{y}_1|\boldsymbol{X}) \cdots P(\boldsymbol{y}_M|\boldsymbol{X})$
RNN-T
$P(\boldsymbol{c}_1|\boldsymbol{X}) \cdots P(\boldsymbol{c}_T|\boldsymbol{X})$
Softmax
$\boldsymbol{g}_{t,u}$
Joint network
$\boldsymbol{p}_u$
$\boldsymbol{h}_t^{\text{enc}}$
Prediction network
Encoder
$\boldsymbol{y}_{u-1}$
$\boldsymbol{x}_1 \cdots \boldsymbol{x}_T$

Attention-based Encoder-Decoder Model

- So far, outputs have always been *time synchronous*
  - “input clock” and “output clock” have a clear relationship defined by the model
  - monotonic relationship between input sequence and output symbols
- AED model removes this relationship, replacing it with *attention* over the inputs determined by the (hidden) state of the decoder.
- All components use neural networks so are end-to-end differentiable.

Attention-based Encoder-Decoder Model

- **Encoder:** Acoustic model using a recurrent network to map acoustic features $X = x_1, \ldots, x_T$ to hidden vectors $h^{\text{enc}} = h_1^{\text{enc}}, \ldots, h_T^{\text{enc}}$.
- **Decoder**: Computes distribution over labels conditioned on previously predicted labels and the acoustics, $P(y_u | y_{u-1}, \ldots, y_0, X)$
- **Attention**: Constructs a *context vector* for the decoder network based on attention weights computed over all frames in the encoder output
- Google's "Listen, Attend, and Spell" model: Chan et al (2016)



$P(\boldsymbol{y}_u|\boldsymbol{y}_{u-1},\ldots,\boldsymbol{y}_0,\boldsymbol{X})$
MLP
$\boldsymbol{h}_u^{\text{dec}}$
$\boldsymbol{c}_u$
Decoder
$\boldsymbol{y}_{u-1}$
$\boldsymbol{h}_{u-1}^{\text{dec}}$
$\boldsymbol{c}_{u-1}$
$\boldsymbol{h}_u^{\text{dec}}$
Attention
$\boldsymbol{h}_t^{\text{enc}}$
Encoder
$\boldsymbol{x}_1$
$\ldots$
$\boldsymbol{x}_T$

The Decoder

- The decoder directly generates the output subword sequence Y
- At each decoding step u, the decoder RNN uses the previous output y_{u-1}, the previous decoder RNN hidden state h_{u-1}^{dec}, and the previous context vector c_{u-1} to generate the current decoder hidden state h_u^{dec}

$$h_u^{\text{dec}} = \text{RNN}(h_{u-1}^{\text{dec}}, y_{u-1}, c_{u-1})$$

- The context vector is computed by the attention mechanism

The Attention Mechanism

- The attention mechanism uses the current decoder RNN hidden state h_u^{dec}, and the sequence of encoder hidden states h_t^{enc} to compute an alignment matrix α_{ut}:

$$\alpha_{ut} = \text{Attention}(h_u^{\text{dec}}, h_t^{\text{enc}})$$

- The alignment vector is used as weights in a weighted sum of the encoder hidden states to compute the context vector c_u:

$$c_u = \sum_{t=1}^{T} \alpha_{ut} h_t^{\text{enc}}$$

- The decoder uses the context vector c_u and the current decoder hidden state h_u^{dec} to estimate the subword distribution:

$$g_u(k) = \exp(MLP(h^{\text{dec}}, c_u))$$

$$P(y = k|u) = \frac{g_u(k)}{\sum_{k'} g_u(k')}$$

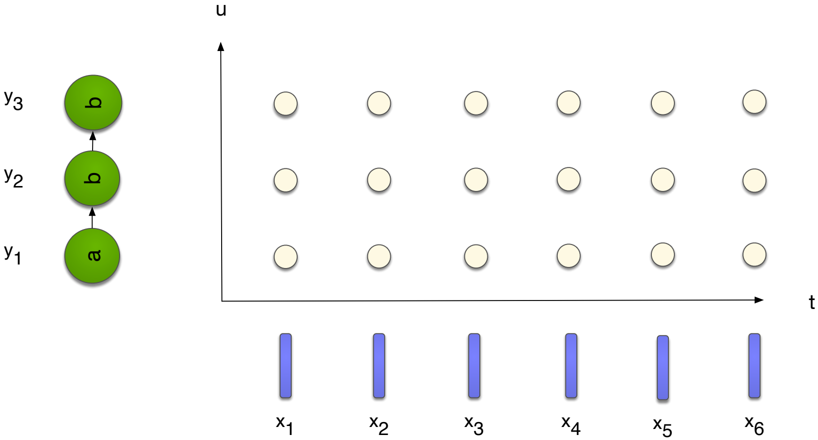
Alignment Vector

- Attention models the alignment between the current output y_u and the input sequence X – it matches the "input clock" with the "output clock"
- Various ways to compute the attention - content-based attention commonly used. Single hidden layer followed by a softmax

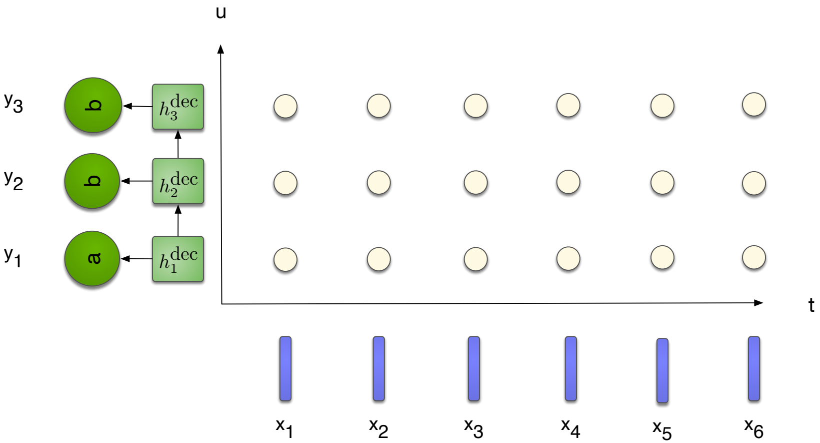
$$e_{ut} = v^T \tanh(W h_u^{\text{dec}} + V h_t^{\text{enc}} + b)$$

$$\alpha_{ut} = \frac{\exp(e_{ut})}{\sum_k \exp(e_{uk})}$$

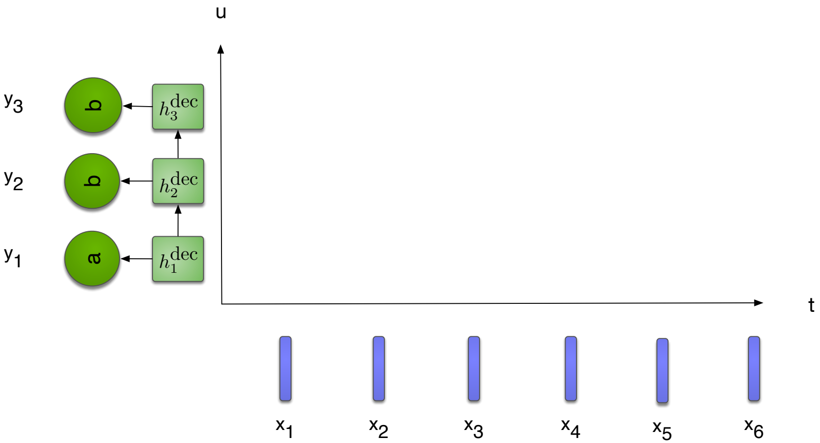
The AED "trellis"



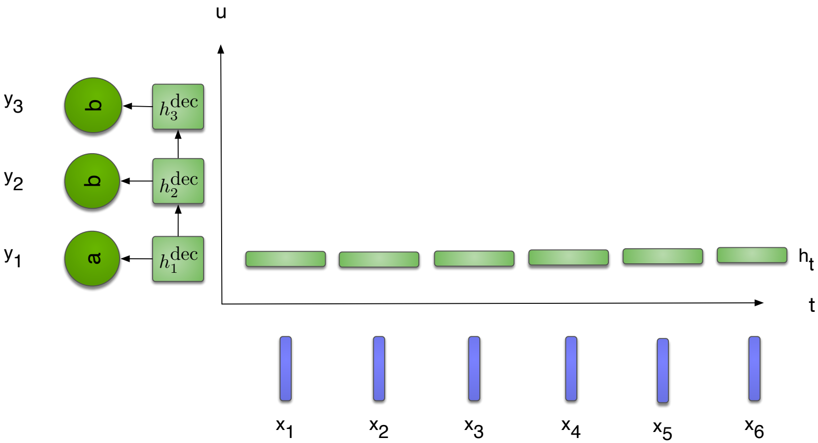
The AED "trellis"



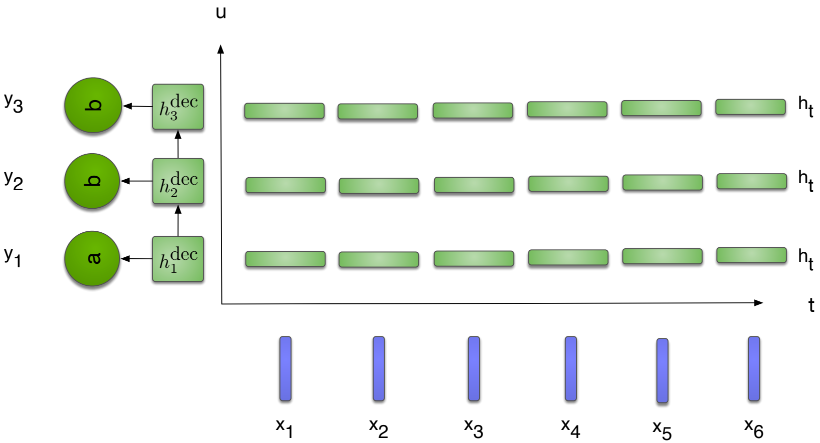
The AED "trellis"



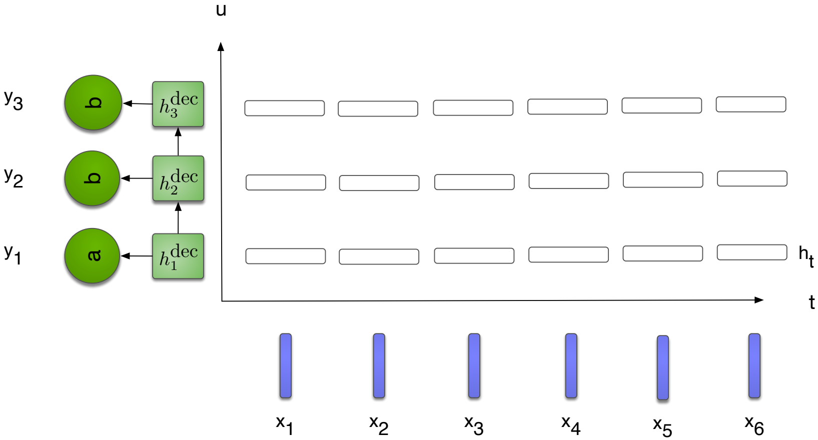
The AED "trellis"



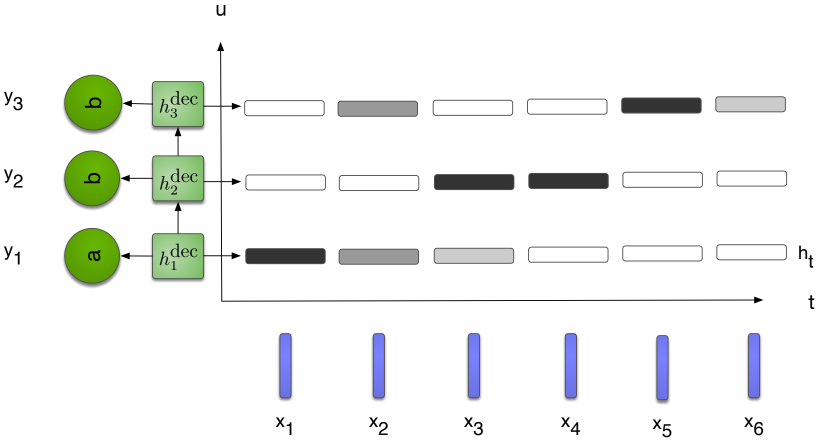
The AED "trellis"



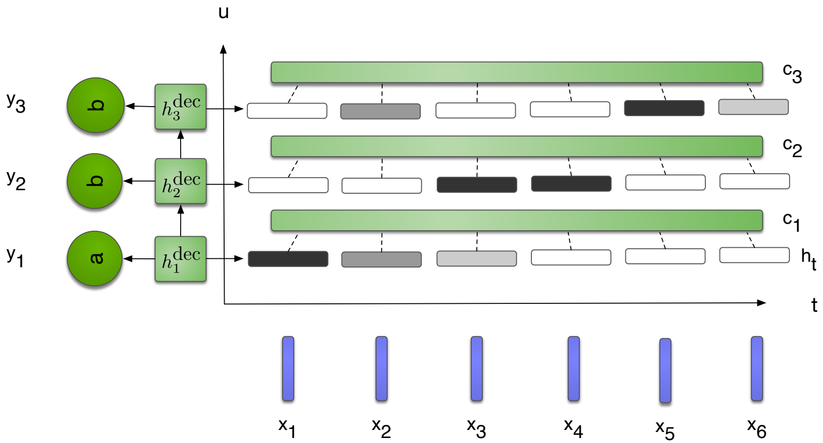
The AED "trellis"



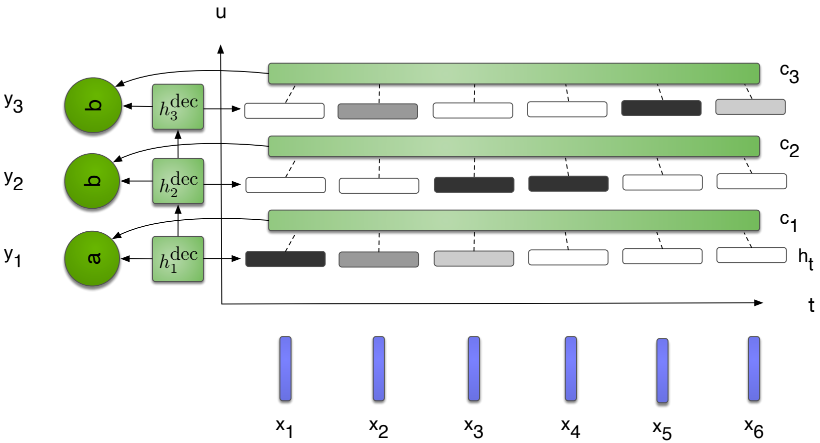
Decoder determines attention over input encodings



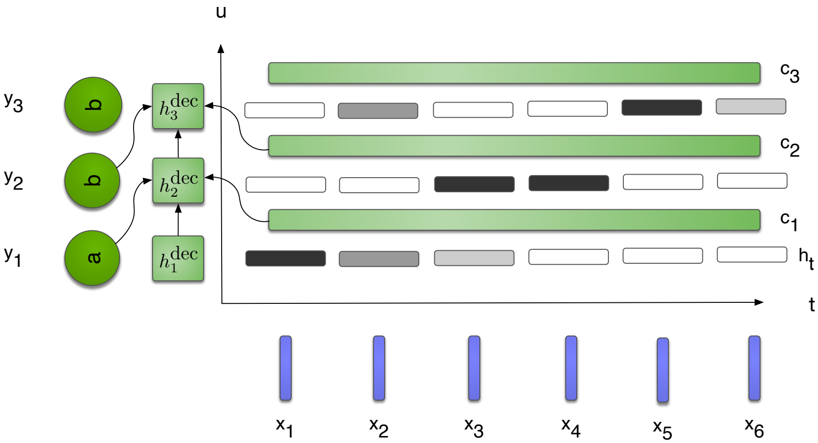
Compute context vector



Generate output unit



Decoder state update



Computing attention

- Attention models the alignment between the current output y_u and the input sequence X – it matches the "input clock" with the "output clock"
- Various ways to compute the attention - content-based attention commonly used. Single hidden layer followed by a softmax

$$
\begin{aligned}
e_{ut} &= v^T \tanh(W h_u^{\text{dec}} + V h_t^{\text{enc}} + b) \\
\alpha_{ut} &= \frac{\exp(e_{ut})}{\sum_k \exp(e_{uk})}
\end{aligned}
$$

Pyramid Encoder

- A significant problem with a naive end-to-end model is the length of the input sequences... A direct BLSTM encoder can be difficult and slow to train – hard to extract the relevant information from many time steps
- Use a pyramid architecture – each successive layer reduces the resolution by a factor of 2.
  - Typical deep BLSTM hidden state (layer j, time t):

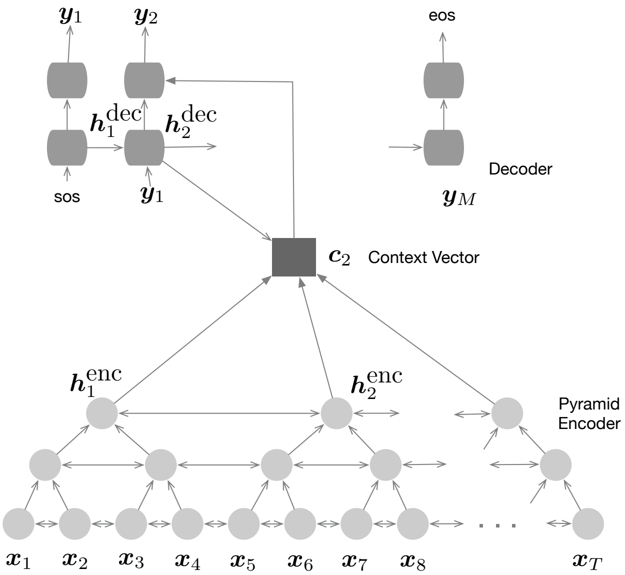
$$h_t^j = RNN(h_t^{j-1}, h_{t-1}^j)$$

  - Pyramid model concatenates consecutive hidden states:

$$h_t^j = pyrRNN([h_{2t-1}^{j-1}, h_{2t}^{j-1}], h_{t-1}^j)$$

  - 3 layers in a pyramid architecture reduces the time resolution (shortens the sequence) by a factor of 8
  - The attention mechanism thus has an easier job, weighting over 8x fewer encoder hidden states

Pyramid encoder example



Learning

- Model trained to maximise the log probability of correct sequences

  $$\sum_u \log P(y_u | X, y_{<u})$$

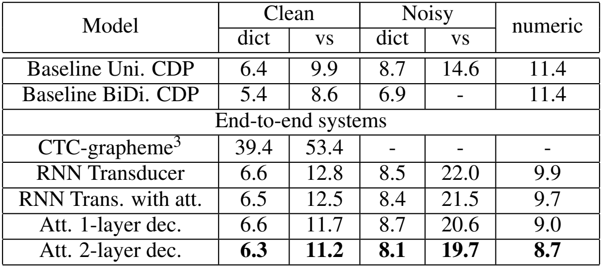
  where $y_{<u}$ is the sequence $y_1, \ldots, y_{u-1}$
- An interesting subtlety: what value should be used for $y_{<u}$?
  - The previous predictions? This is consistent between training and test, but adds noise at training time
  - The ground truth labels (*teacher forcing*)? This speeds up learning, especially early on, but there is a mismatch between training and testing
  - **Scheduled sampling**? Sample a label from the estimated distribution. This reduces the noise in training, but doesn't create a big gap between training and test

Decoding and Rescoring

- Decode without a separate pronunciation model or an external language model!
- Simply decode the grapheme sequence! (It is possible to rescore with a language model if desired)
- Decoding uses a beam search in which *n*-best hypotheses are retained at each decoding step

Results (2017)

Google Voice Search data, 12,500h training data, 15M hand-transcribed utterances



Model	Clean		Noisy		numeric
	dict	vs	dict	vs	
Baseline Uni. CDP	6.4	9.9	8.7	14.6	11.4
Baseline BiDi. CDP	5.4	8.6	6.9	-	11.4
End-to-end systems					
CTC-grapheme[3]	39.4	53.4	-	-	-
RNN Transducer	6.6	12.8	8.5	22.0	9.9
RNN Trans. with att.	6.5	12.5	8.4	21.5	9.7
Att. 1-layer dec.	6.6	11.7	8.7	20.6	9.0
Att. 2-layer dec.	**6.3**	**11.2**	**8.1**	**19.7**	**8.7**

Prabhavalkar et al (2017)

Other Refinements

- Wordpiece models – rather than using single graphemes as labels use multi-grapheme units (up to a word in length) - similar to bye pair encoding in machine translation
- Multiheaded attention – use multiple attention distributions
- Minimum WER training – modify the loss function to interpolate a word error rate term
- Label smoothing – smooth the ground truth distribution by interpolating with a uniform distribution
- LM rescoring – use an external language model (5-gram) trained on large amount of text

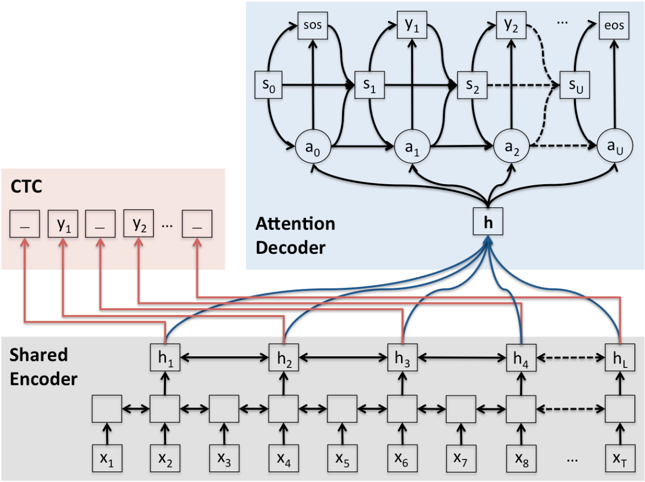
Reduced WER on Voice Search from 9.2% to 5.6% – their hybrid HMM-LSTM system has WER of 6.7% on this task

Chiu et al (2018)

Hybrid CTC/Attention

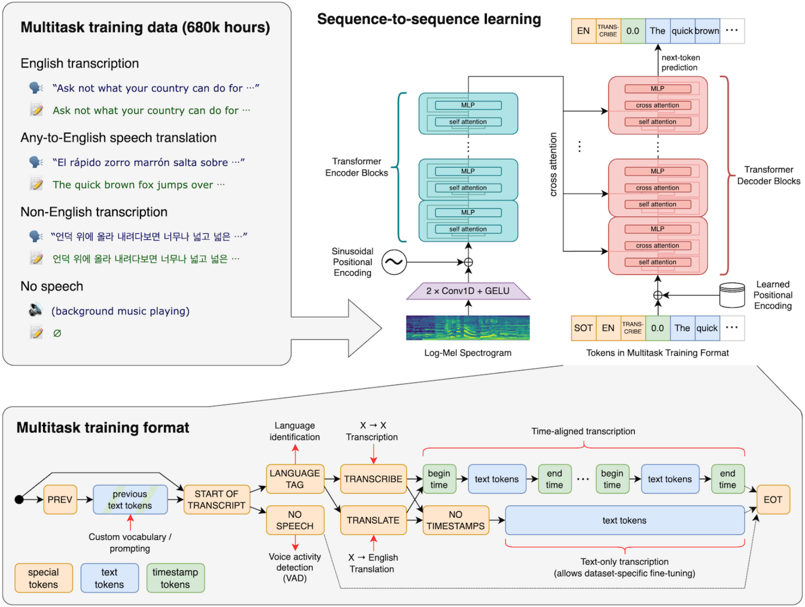
- Attention is very flexible – does not constrain relationship between acoustics and labels to be monotonic
- This can be a problem, especially when huge amounts of training data not available
- Possible solutions:
  - Windowed attention, in which the attention is restricted a set of encoder hidden states
  - Hybrid CTC/Attention model - use CTC and attention jointly during training and recognition – regularises the system to favour monotonic alignments

Hybrid CTC/Attention



Watanabe et al (2017)

Whisper: an open AED model



Multitask training data (680k hours)

English transcription

"Ask not what your country can do for ···"

Ask not what your country can do for ···

Any-to-English speech translation

"El rápido zorro marrón salta sobre ···"

The quick brown fox jumps over ···

Non-English transcription

"언덕 위에 올라 내려다보면 너무나 넓고 넓은 ···"

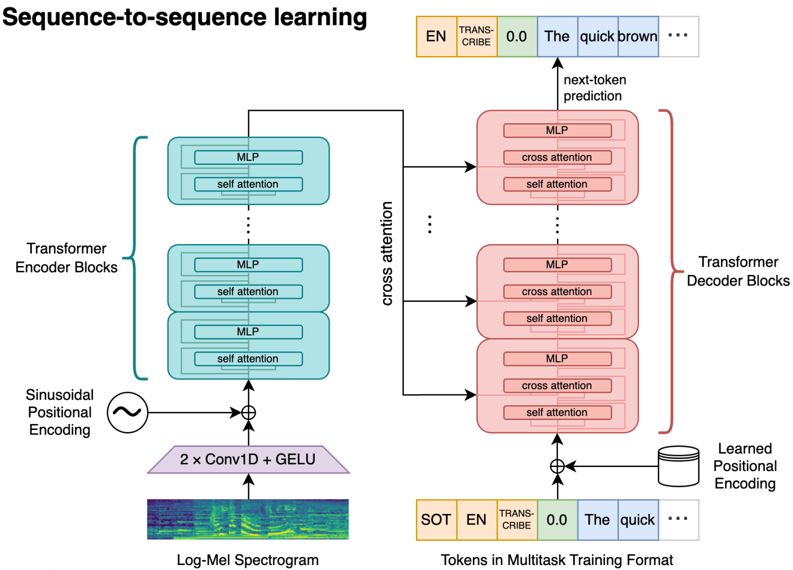
언덕 위에 올라 내려다보면 너무나 넓고 넓은 ···

No speech

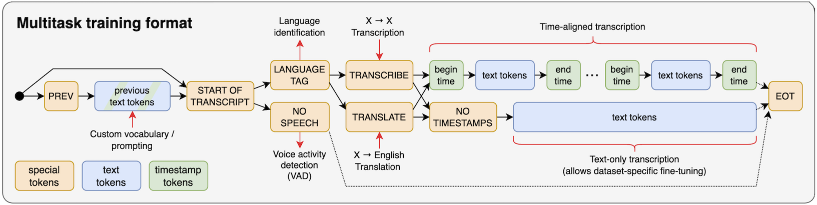
(background music playing)

∅

Whisper: an open AED model



Whisper: an open AED model



Summary

- End-to-end models for speech recognition: CTC, RNN Transducer, Attention Encoder-Decoder
- RNN Transducer and Attention-based model integrate acoustic model, pronunciation model, and language model into a single neural network
- With large amounts of hand-transcribed training data, attention-based model can be more accurate than context-dependent NN/HMM
- Attention based model operates over an utterance at a time (since attention is over the complete encoded utterance)
- Remains an active research area! Eg. recent use of self-attention (Transformer) in place of recurrent architectures

Reading

- Watanabe et al (2017), "Hybrid CTC/Attention Architecture for End-to-End Speech Recognition", IEEE STSP, 11:1240–1252. https://ieeexplore.ieee.org/document/8068205
- Chan et al (2016), "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition", IEEE ICASSP, pp. 4960-4964 https://ieeexplore.ieee.org/abstract/document/7472621
- Chiu et al (2018), "State-of-the-art sequence recognition with sequence-to-sequence models", IEEE ICASSP. https://arxiv.org/abs/1712.01769
- Prabhavalkar et al (2017), "A Comparison of Sequence-to-Sequence Models for Speech Recognition", Interspeech. https://www.isca-speech.org/archive/Interspeech_2017/abstracts/0233.html