MPhil in Machine Learning and Machine Intelligence

Module MLMI2: Speech Recognition

L16: End-to-End Trainable Systems

Phil Woodland
pcw@eng.cam.ac.uk

Michaelmas 2021

Cambridge University Engineering Department

Introduction

This lecture discusses the use of End-to-End Trainable Speech Recognition Systems.

It includes

- Motivation
- Types of approach:
  1. CTC models
  2. RNN-Transducers
  3. Attention-based Encoder-Decoder Models

Various types of architectural decisions will be discussed including

- Combination of attention and conventional hybrid systems (source-channel models)
- incorporation of language models
- Model structures including transformer models

We will also discuss some practical issues that are required to improved performance.

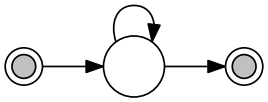
Motivation

Advances in deep learning has led to increasing interest in **end-to-end** ASR systems

- Key attribute is that they are end-to-end **trainable**
- Ideally **not factorised** into different model components (acoustic model with alignments, pronunciation model, decision trees, language model)
- Should be **simple** to train/deploy
- Only one set of training data (need to learn everything from it)
  - input speech (raw waveform or mel filterbank)
  - output words (or characters) or word-pieces
- Some of these models are now trained on many 1000s (or 10s of 1000s) of hours of audio
- Initially interest in Connectionist Temporal Classification (CTC)
  - train a neural network discriminative model without needing explicit alignment
- Lattice-free MMI can also be viewed as end-to-end (by some people) since no alignment
- Interest has shifted to **encoder-decoder** models (with **attention**)
  - but there are issues with latency
- CTC has been extended to the RNN-Transducer (RNN-T) model which is more suitable for streaming applications
- Many systems in practice still use a separate (additional) **language model**

Low Frame Rate Systems

- Conventional hybrid systems often using lower frame rate in commercial systems
- Reduce from 10 ms to 30 ms
- Modify topology from 3 state left-to-right to single emitting state



- 30ms input can use decimation (1 frame in three) or (more normally) stacking three adjacent frames at input

This type of approach is widely used:

- for all end-to-end trainable systems i.e. CTC, RNN-T, attention-based systems
- also used for LF-MMI

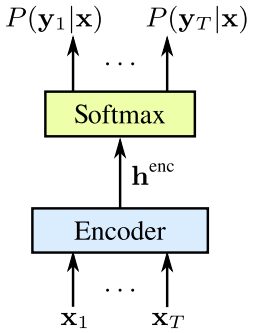
Use in direct sequence-to-sequence systems to **reduce the length of the input sequence**

- Fewer separate time frames / probability calculations
- Smaller ratio between the input length and the output length

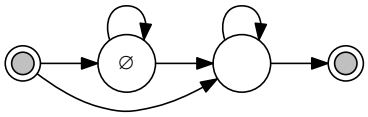
Connectionist Temporal Classification (CTC)

CTC became popular since it does not require explicit alignments in training.

- Augments the set of output symbols, y_t, originally characters with a **blank** symbol
- Uses a compressive mapping to remove repeated symbols and remove blanks
- Trained to directly maximise the posterior probability of the mapped output sequence (summing over all valid alignments)
- Typically uses a deep LSTM encoder model



- Can view "blank" symbol / emitting state shared among sub-word models
- Equivalent HMM model shown



- Originally used with graphemic sub-word units (but could be words or word-pieces)
- Forward-backward algorithm can be used to sum over all possible alignments

Advantages / Disadvantages of CTC

Key advantages of CTC are those already highlighted:

- Does not require explicit frame-level alignments in training
- Forward-backward algorithm can be used to sum over all possible alignments
- Introduces CTC sequence-level objective function

Furthermore, due to structure, ensures that the usual monotonic relationship between input and output is preserved.

It is found that CTC trained models automatically learn alignments well and have a very sharp output distribution.

However there are several issues:

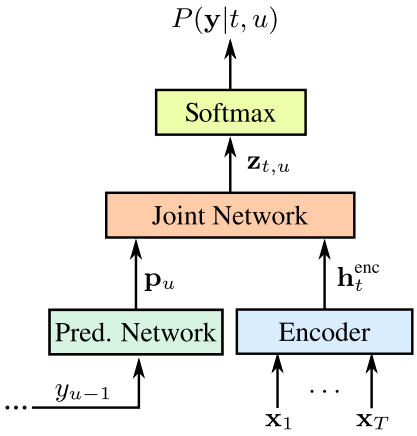
- CTC doesn't enforce any relationship between the outputs: effectively assumes they are independent
- Require additional models such as lexicon and language models to introduce dependencies
- end-to-end training on sequence level objective only updates the acoustic model, other models are not altered.

RNN Transducer Model

- **RNN-T** aims to overcome issues with CTC.
- Retains blank symbol (but treated differently to CTC: repeated outputs don't need to be separated by blank)

It includes

- Encoder model (deep LSTM) similar to acoustic model
- Prediction network (a recurrent network): takes previous (non-blank) sub-word output symbol and computes $\boldsymbol{p}_u$ dependent on all outputs to that point. Similar to LM.
- Prediction and Encoder outputs combined by joint network (shallow feed-forward network)
- softmax output gives distribution over output symbols
- If output blank, move onto next frame, otherwise make new prediction
- auto-regressive decoding (and can include beam-search maintaining multiple histories)



- Can operate in a left-to-right manner with low latency if encoder is uni-directional
- Can perform well with a large amount of training data
- Can give good performance with small footprint (Google on-device model)
- Can reduce the output history length fed back & merge paths (& hence lattices)

Nature of Targets

Current trend for discriminative models is to use

- **graphemic** i.e. character targets
- **word-piece** targets

Key requirement is that there is a direct mapping to words (& hence can be end-to-end)

- Don't normally use words directly as too many
- Also issues in difference between input rate (e.g. 30 ms) and output rate (characters)

If use **graphemes** (characters)

- no need to have a lexicon (hence no OOVs)
- can perform ok, but much better if incorporate an external language model

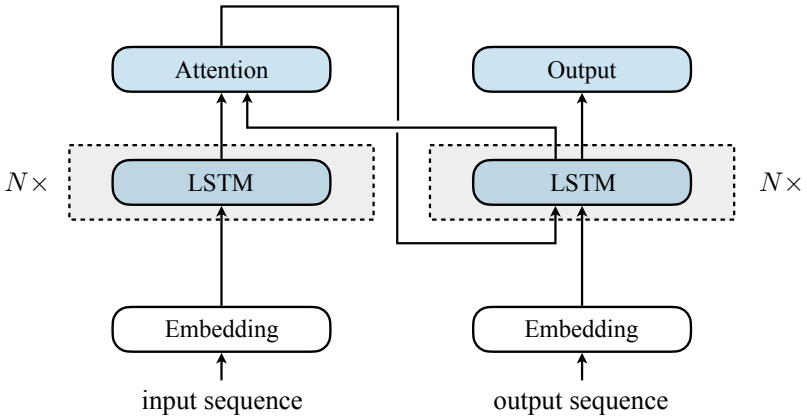
If use **word-pieces**

- Can also get full coverage of words (word-pieces normally marked differently if include word-boundaries or word-internal)
- various approaches to determine set of word-pieces (including **byte pair encoding**)

Encoder-Decoder Models

Directly model the relationship of the output sequence to the input sequence

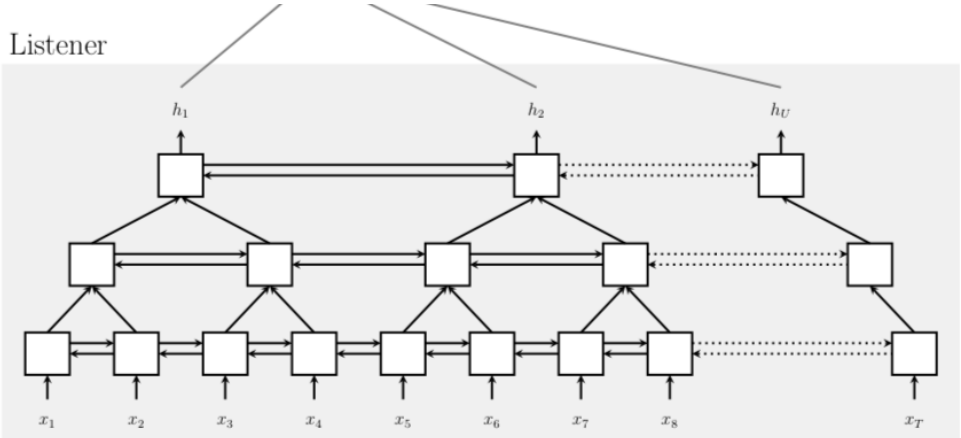
- encoder produces fixed length vectors for each input time
- typically produced by an LSTM or bi-directional LSTM
- uses **attention** to determine the encoder vector weighting
  - how the input at position τ is weighted in producing the output at position i



- Decoder relies on the input of the attention-weighted encoder and **previous output**
  - generates probability distribution over next output symbol
  - autoregressive decoder

Pyramidal LSTMs

- Issues between very different input and output rates
  - attention can wander
  - use **pyramidal LSTM** in encoder "Listener"
  - Can be viewed as optimised time decimation



Listen, Attend and Spell

- LAS is an attention-based encoder-decoder architecture
- uses multi-grapheme word-piece output targets
- minimum word error rate training (not just posterior probability)
- **multi-head attention**
  - multiple attention mechanisms are used in parallel
  - attention focuses on different aspects of the signal

- normally correct output sequence (teacher-forcing) used in training
- **scheduled sampling** used to replicate effect in inference/decoding of incorrect outputs
- in training interleave the correct values with those produced by the model
- amount of model produced output depends on the epoch of training

- For inference/decoding, normally a number of output hypotheses are maintained in parallel
- Known as **beam-search** where the beam is the number of alternatives
- improved performance over simple greedy search at increased computational cost

External Language Models for LAS systems

A number of approaches have been used:

- **Shallow fusion** (or joint decoding) Use combination of language model score (log probability) and LAS model "score"

$$\log P(y_t) = \log P_{\text{Att}}(y_t) + \beta \log P_{\text{LM}}(y_t)$$

  where $\log P_{\text{Att}}(y_t)$ is LAS model score, and $\log P_{\text{LM}}(y_t)$ is external LM score, β is LM scaling factor. Note that need to deal with potential word level LM and sub-word level LAS system output.
- **Deep fusion** Input to the final softmax layer concatenates LAS output hidden states and the external LM output hidden states. The LAS model is independently trained before fusion. External language model can be jointly fine-tuned with the LAS model.
- **"Cold" fusion**: Output distribution from external LM transformed into a vector which is concatenated with the LAS output hidden state, similar to the deep fusion. The external LM parameters are fixed. The LAS model is trained from scratch with external LM information.

The use of external LMs are potentially trained on large corpora

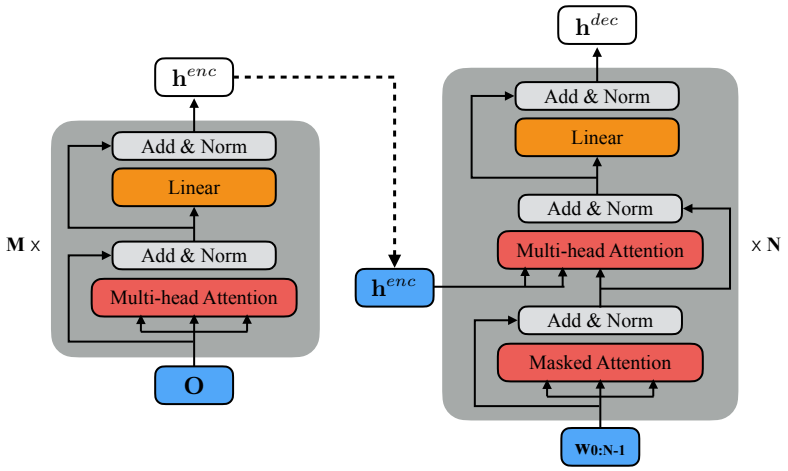
- Important as otherwise system only trained on the available audio data

Transformer Models for ASR

Transformer models can be used in place of LSTMs in the encoder-decoder architecture.

Encoder (shown left)

Decoder (right)



- Transformer encoder. Input sequence is $\boldsymbol{O} = [\boldsymbol{o}(1), \boldsymbol{o}(2), \cdots, \boldsymbol{o}(T)]$. M, N represents the number of blocks. Encoder outputs a sequence of length T of hidden states $\boldsymbol{h}^{\text{enc}}$
- Transformer decoder. Input are encoder hidden states and word embedding sequence $w_{0:N-1}$. Output of decoder is another sequence of hidden states $\boldsymbol{h}^{\text{dec}}$ which yields distribution of next words in sequence.
- Absolute positional encodings representing the position of each token in the sequence can be directly added to input vector sequences for both encoder and decoder.

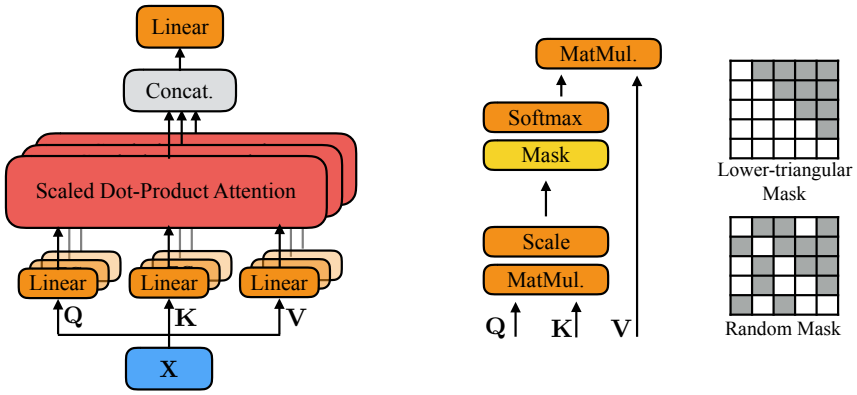
Attends over complete input sequence and output sequence up to current word.

Attention and Masked Attention

Transformer uses self-attention throughout.

Multihead attention with 3 attention heads (left below)

- Q, K, V represent the query, key and value in the scaled dot-product attention
- Computed from linear projection of the input X using different learnable projection matrices.



Masked attention where a sequence length of 5 is shown for illustration (right above).

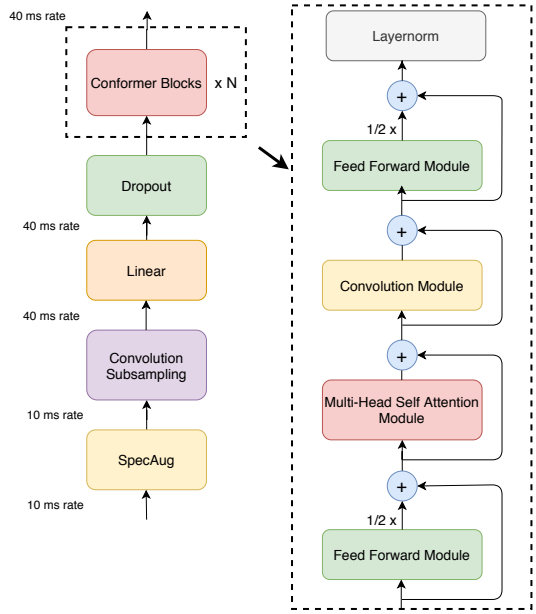
- Lower-triangular mask and random mask are shown, where shaded blocks are masked.
- In the decoder, a lower-triangular mask is used so that future words not included in the computation

Conformer Encoder

Conformer (adds **convolution** to transformer) is the current state-of-the-art encoder.
It can be used with both

- encoder-decoder architectures and
- neural transducers (conformer transducer): needs suitable mask attention for streaming

- input embedding (convolution & sub-sampling)
- stack of repeated conformer blocks (right-hand diagram)



Each confomer block has

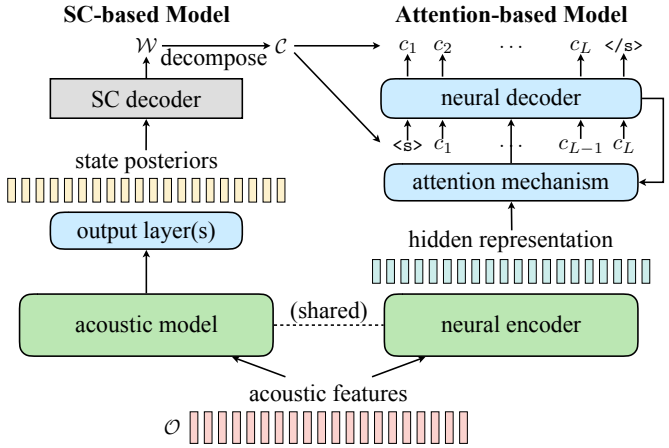
- Feed-forward module ($\times 0.5$)
- multi-headed self-attention module including relative positional encoding
- Convolutional module (itself composed of several sub-components)
- 2nd feed-forward module ($\times 0.5$)
- Use of residual connections as shown
- Layer normalisation

Figure from conformer paper (Gulati et al, 2020), which also gives details of blocks and training.

Combining Attention and Hybrid Models

Attention and hybrid (or source-channel) models are very different and can be combined in a number of ways.

- Illustration of the "Integrated Source Channel and Attention" (ISCA) architecture
- Hypotheses from the source-channel model are fed into the attention-based model and scored by attention system (which operates on sub-word units)
- Uses N-best framework (but only need to compute encoder once)



This type of architecture can reduce word error rate at the cost of complexity.

- If acoustic model is shared less improvement
- Illustrates very different models and scope for combination
- Can produce state-of-the-art performance if conformer encoder is used as well as a high-performance DNN-HMM (e.g. LF-MMI TDNN).

Summary

End-to-end trainable models have many attractive features and are very general

Described

- CTC/RNN-T model
- Encoder-Decoder approaches using attention (recurrent models and transformer)
- In theory just train model and decode
  - use scheduled sampling
  - need to add extra decoding parameters (coverage penalties based on attention, word-insertion penalty)

Much research focus has switched to end-to-end trainable models for both

- RNN-T models including field deployments (small footprint, low latency)
- attention based encoder-decoder models

In both cases there are many avenues being investigated

- Reduced latency for encoder-decoder models
- Performance on rare / unseen words in acoustic training
- Use where more limited training data (transfer learning e.g. wav2vec 2.0)
- Extracting appropriate confidence scores
- Noise robustness / adaptation (currently use data augmentation techniques)
- Development of many alternative forms e.g. further variants of basic transformer approach

References (partial list)

- D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel & Y. Bengio, “End-to-end attention-based large vocabulary speech recognition”. Proc. ICASSP, 2016.
- A. Baevski, H. Zhou, A. Mohamed & M. Auli, “wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations”, Proc. NeurIPS, 2020.
- C.C. Chiu, T.N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen,Z. Chen, A. Kannan, R.J. Weiss, K. Rao, E. Gonina, N. Jaitly,B. Li, J. Chorowski, & M. Bacchiani,“State-of-the-art speech recognition with sequence-to-sequence models”, Proc. ICASSP, 2018.
- W. Chan, N. Jaitly, Q.V. Le, & O. Vinyals, “Listen, Attend and Spell: A neural network for large vocabulary conversational speech recognition”, Proc. ICASSP, 2016
- L. Dong, S. Xu, & B. Xu, “Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition”, Proc. ICASSP, 2018.
- J.K. Chorowski, D. Bahdanau, D. Serdyuk, K.Cho, & Y. Bengio, “Attention-Based Models for Speech Recognition”, Proc. NIPS, 2015.
- M. Ghodsi, Xi. Liu, J. Apfel, R. Cabrera & Eugene Weinstein, “RNN-Transducer with Stateless Prediction Network”, Proc. ICASSP, 2020.
- A. Graves, S. Fernandez, F. Gomez, & J. Schmidhuber, “Connectionist Temporal Classification: Labeling Unsegmented Sequence Data with Recurrent Neural Networks”. Proc. ICML, 2006.

- A. Graves, A. Mohamed, & G. Hinton, “Speech Recognition with Deep Neural Networks”, in Proc. ICASSP, 2013.
- A. Gulati, J. Qin, C-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, R. Pang, “Conformer: Convolution-augmented Transformer for Speech Recognition”, Proc. Interspeech, 2020.
- Q. Li, C. Zhang & P.C. Woodland, “Integrating source-channel and attention-based sequence-to-sequence models for speech recognition”, Proc. IEEE ASRU Workshop, 2019
- Q. Li, C. Zhang & P.C. Woodland, “Combining Frame-Synchronous and Label-Synchronous Systems for Speech Recognition”, arXiv:2107.00764, 2021
- R. Prabhavalkar, K. Rao, T. Sainath, B. Li, L. Johnson, N. Jaitly, “A Comparison of Sequence-to-Sequence Models for Speech Recognition”, Proc. Interspeech, 2017
- R. Prabhavalkar, Y. He, D. Rybach, S. Campbell, A. Narayanan, T. Strohman, T. Sainath, “Less Is More: Improved RNN-T Decoding Using Limited Label Context and Path Merging”, Proc. ICASSP 2021.