AdvNLP Week 7

Machine Translation

Dr Julie Weeds, Spring 2024

Warm-up

Compare the phrases on the left with the phrases on the right…

- panda car
- memory lane
- rocket science
- crash course
- rat race

- car park
- climate change
- application form
- student house
- bank account

Previously

- Distributional semantics
- Language models
- Neural language models
- Sequence labelling
- Sequence classification

Overview

- What makes machine translation (MT) hard?
- Evaluation of MT
- Classical MT (Pre 1990s)
- Statistical MT (1990-2015)
  - Word-based models
  - Phrase-based models
- Neural MT (2015 -)
  - Encoder-decoder models

MT Lecture Questions

1. What makes Machine Translation a hard problem?
2. What aspect of MT can be evaluated by monolingual raters and what aspect requires bilingual raters?
3. What do BLEU and chrF have in common? How are they different?
4. What are some of the key components / choices in setting up a statistical MT system?
5. Why should neural MT work better?

Why is/was MT hard?

- Lexical differences
- Structural differences (morphological differences and syntactic differences)
- Study of systematic cross-linguistic similarities and differences is called **linguistic typology**
  - See World Atlas of Language Structures (Dryer and Haspelmath, 2013)

BLEU

- computes modified precision for unigrams, bigrams, trigrams and often quadrigrams
- combines using geometric mean
- incorporates a penalty for translations which are too short
- good for evaluation of incremental changes to same general architecture
- see Papineni 2002

chrF

- chrP = percentage of character 1-grams, 2-grams, ... , k-grams in the hypothesis that occur in the reference, averaged
- chrR = percentage of character 1-grams, 2-grams, ..., k-grams in the reference that occur in the hypothesis, averaged

$$chrF\beta = (1+\beta)^2 \frac{chrP\,.\,chrR}{\beta^2.\,chrP + chrR}$$

β=2 gives twice as much weight to chrR as to chrP

Key points in statistical MT

- focus on the result NOT the process
- based on probabilities derived from *parallel corpora*
- Estimation maximization to obtain word translation probabilities
- Alignment models e.g. word alignment vs phrase alignment
- Generative vs discriminative models
- Decoding is a search problem
- Inability to generalise

Part 3

Neural Machine Translation

Neural Machine Translation (NMT)

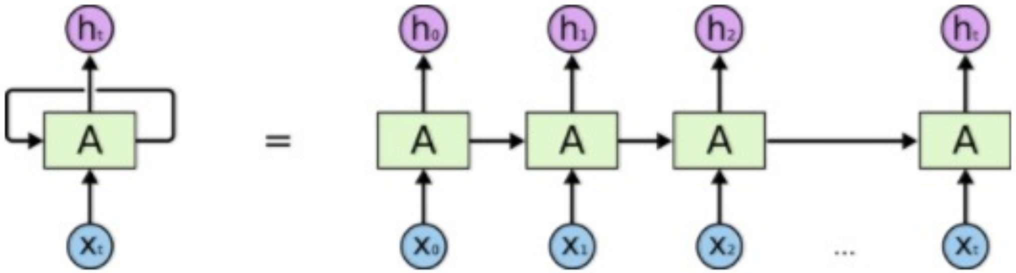
- Continuous representations (e.g., word2vec embeddings) for words and phrases are able to capture their morphological, syntactic and semantic similarity
- As in SMT, train on parallel corpora where sentences are aligned
- Maximise the probability of the sequence of tokens in the target language $y_1 \ldots y_m$ given the sequence of tokens in the source language $x_1 .. x_n$

$$P(y_1, \ldots, y_m | x_1, \ldots, x_n)$$

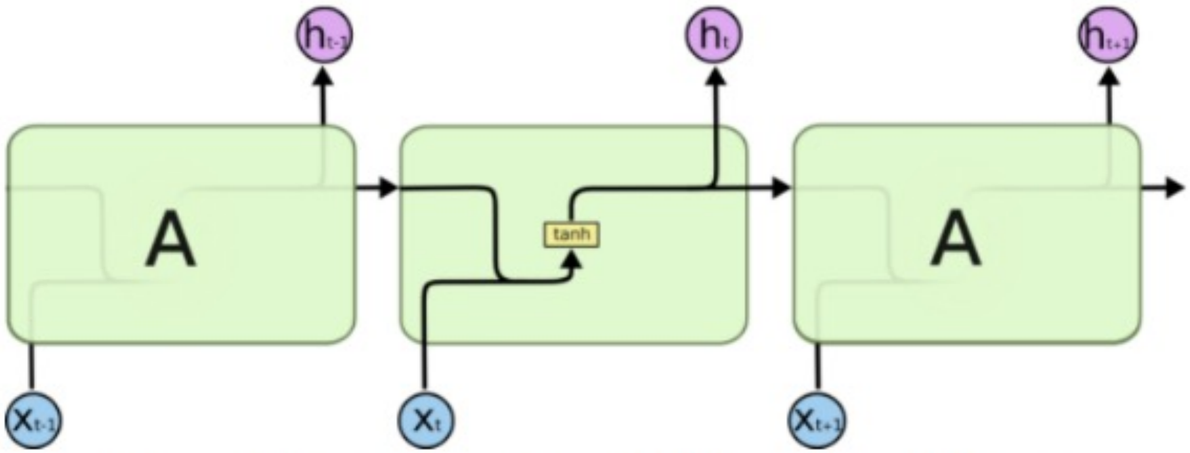
Basic architecture for NMT

- Encoder – decoder architecture
  - Aka sequence-to-sequence or seq2seq architecture
- 2 recurrent neural networks (RNNs) – one to consume the input text sequence and one to generate translated output text.

RNNs



An unrolled recurrent neural network.



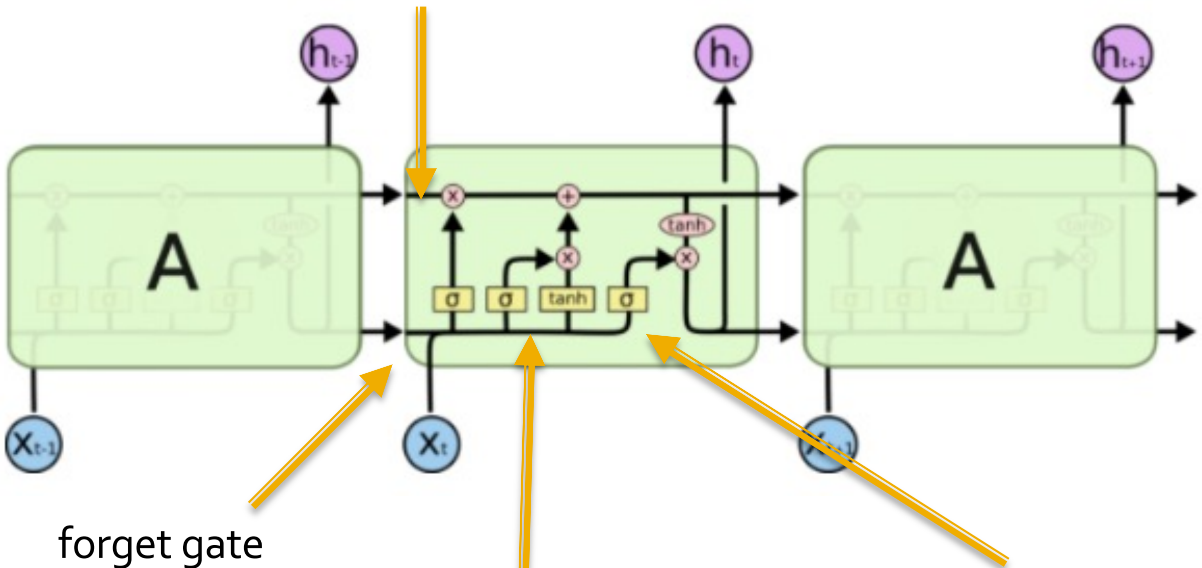
The repeating module in a standard RNN contains a single layer.

RNNs are very effective at learning language models i.e., P(E) the probability of a sentence in a given language. During training, the error (i.e., difference between output and next word) is back-propagated to update the weights used to combine X_t and h_{t-1} AND the representations of the words (X_t)

Long short term memory networks (LSTMs)

- Simple RNNs struggle with long term dependencies e.g., “He grew up in Spain. He speaks fluent …”

cell state (long term memory)



LSTMs overcome this problem by having 4 interacting layers in each repeated module.

forget gate (what to forget)

input gate (which values to update and what with)

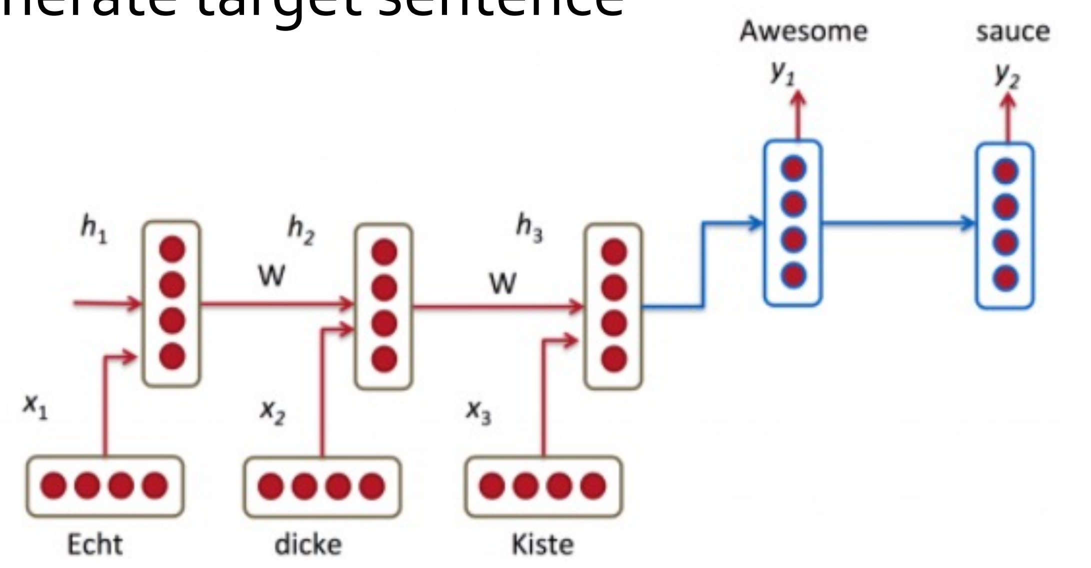
output gate (what to output)

Basic architecture for NMT

- RNN1, the encoder, builds a representation of the source sentence $x = x_1 \ldots x_n$

$$h = encoder(x)$$

- The output from RNN1 (after the complete source sentence has been read) is input to RNN2, the decoder to generate target sentence



$$y_{i+1} = decoder(h, y_1 \ldots y_i)$$

Encoder-decoder details

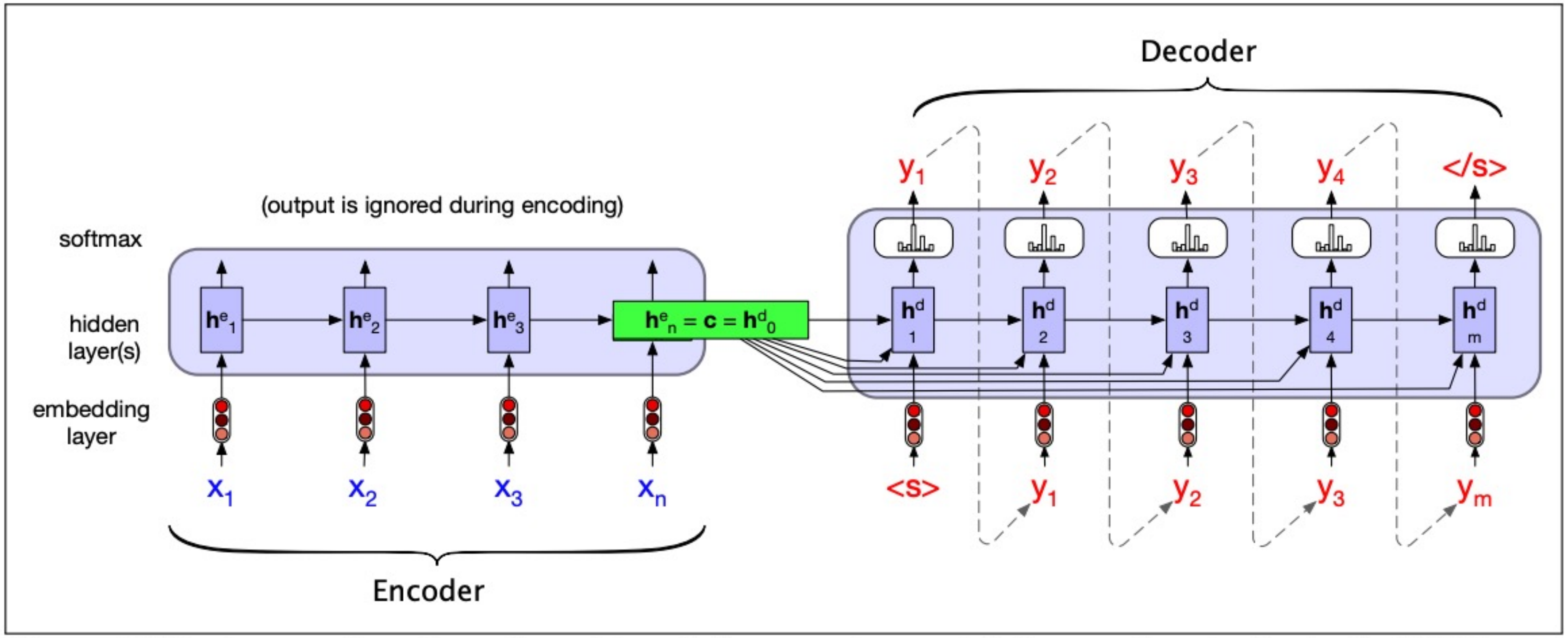


Figure 9.18 A more formal version of translating a sentence at inference time in the basic RNN-based encoder-decoder architecture. The final hidden state of the encoder RNN, h_n^e, serves as the context for the decoder in its role as h_0^d in the decoder RNN, and is also made available to each decoder hidden state.

Possible weaknesses

- Slow training and inference speed
- Ineffectiveness at dealing with rare words
- Output sentences that do not translate all words of the input sentence
- Difficulty in translating long sentences since the encoder output (or context) needs to encode the whole sentence
  - Information from start of sentence may be lost

Rare words (Luong et al. 2015)

- Due to computational constraints, NMT systems usually limited to top 30K-80K of most frequent words in each language
- Unknown/rare words can be translated using a dictionary or exact copy provided it is known which source word generated UNK token in target.
- Problem when sentence contains multiple rare words
- Luong et al. first use a word alignment of parallel corpora and annotate unknown words with positional information (e.g., UNK1)
- Output from NMT can then be post-processed

Subword tokenization

- Word vocabulary is huge and sparse
- Character vocabulary is small and dense, but lacking in semantic meaning
- Subword tokenization provides a compromise
- Frequent words tend to be a token whereas rare words will be broken down into subwords based on character n-grams
- Shared vocabulary for source and target languages – makes it easy to copy tokens like names from source to target
- Common algorithms include
  - BytePiece Encoding (BPE)
  - Wordpiece algorithm
  - Unigram / SentencePiece algorithm

Long sentences

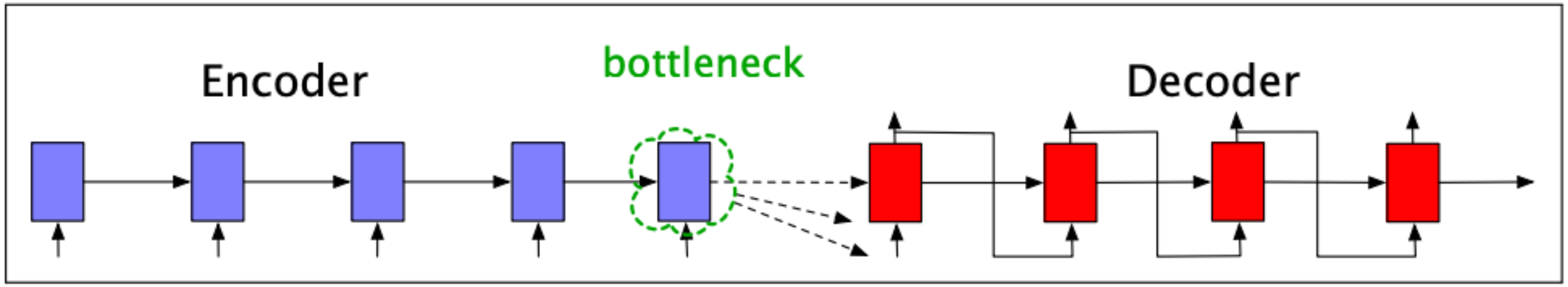


Figure 9.20 Requiring the context c to be only the encoder's final hidden state forces all the information from the entire source sentence to pass through this representational bottleneck.

- Attention (more on this next week) provides a way for the decoder to get information from all of the hidden states of the encoder rather than just the last hidden state

Attention

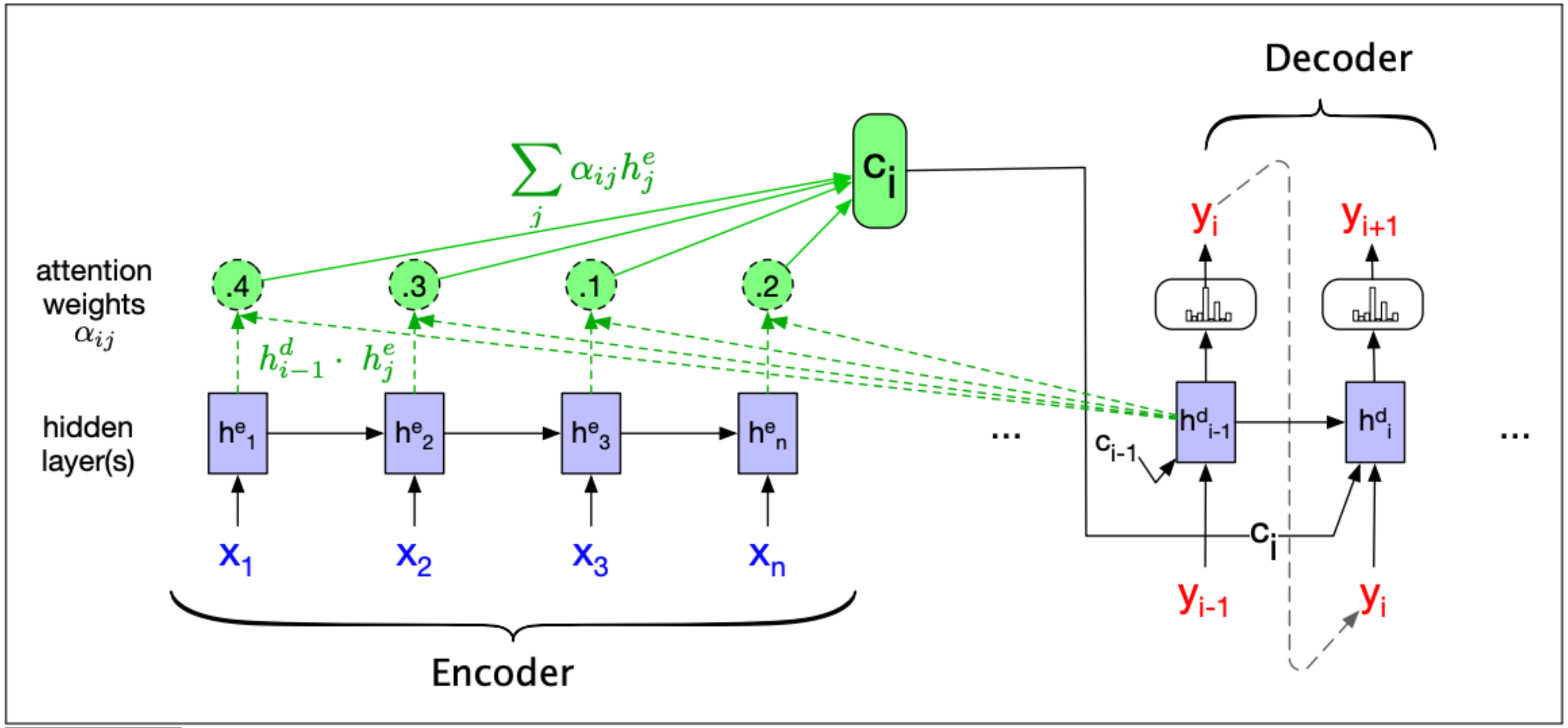
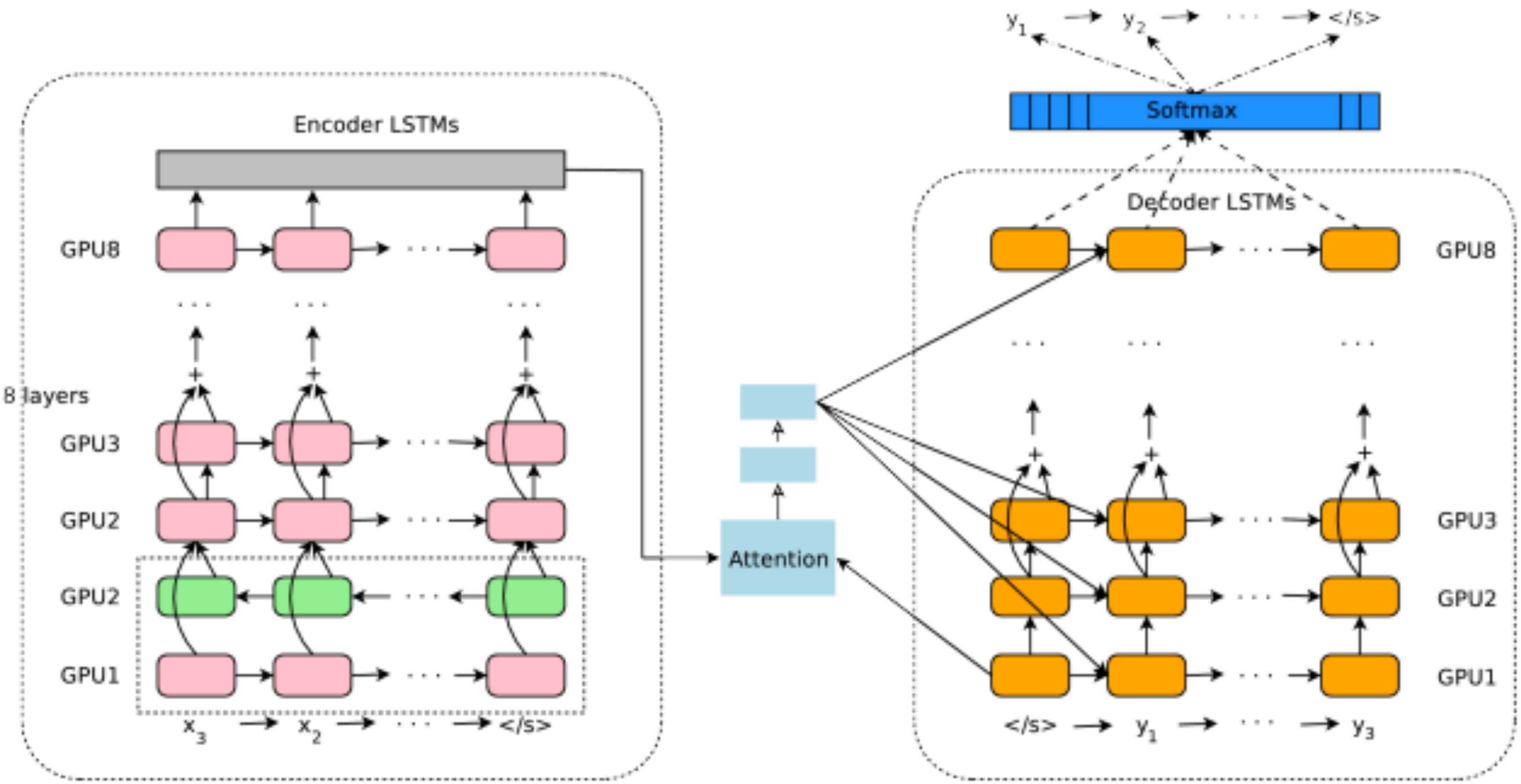


Figure 9.22 A sketch of the encoder-decoder network with attention, focusing on the computation of $\mathbf{c}_i$. The context value $\mathbf{c}_i$ is one of the inputs to the computation of $\mathbf{h}_i^d$. It is computed by taking the weighted sum of all the encoder hidden states, each weighted by their dot product with the prior decoder hidden state $\mathbf{h}_{i-1}^d$.

Google NMT (GNMT)

- Recurrent networks are LSTMs with attention (8 layers - residual connections between layers to encourage gradient flow)
- For parallelism, the attention from the decoder network connect to top layer of encoder network
- Low-precision arithmetic for inference, accelerated using special hardware (Google's TPU)
- Rare words dealt with using sub-word units (balancing flexibility of single characters with efficiency of full words)
- Beam search techniques includes a length normalization procedure and a coverage penalty to encourage model to translate all of the input

GNMT Architecture



Transformers and LLMs in MT?

- Transformers generally have higher performance than LSTMS and GRUs
- Generally replacing seq2seq architectures
- More on this in weeks 8-10

- https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9355969
- https://arxiv.org/abs/2209.07417

Open questions

- High resource vs low resource languages

Evaluation Exercise

- What are the chrF1 and chrF2 scores for each of the following hypothesis translations if k = 2?

REF	witness for the past,	Unigram precision	Unigram recall	Bigram precision	Bigram recall	ChrF1	ChrF2
HYP1	witness of the past,						0.86
HYP2	past witness						0.62

References

- Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models, In *EMNLP*
- Philip Koehn. 2004. Pharoah: A Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In *AMTA*
- Minh-Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals and Wojciech Zaremba. 2015. Addressing the Rare Word Problem in Neural Machine Translation. In *ACL*
- Ilya Sutskever, Oriol Vinyals and Quoc Le. Sequence to Sequence Learning with Neural Networks. In *NIPS*
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc Le and Mohammad Norouzi. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. ArXiv Oct 2016