COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

The Attention Mechanism & Encoder-Decoder Variants

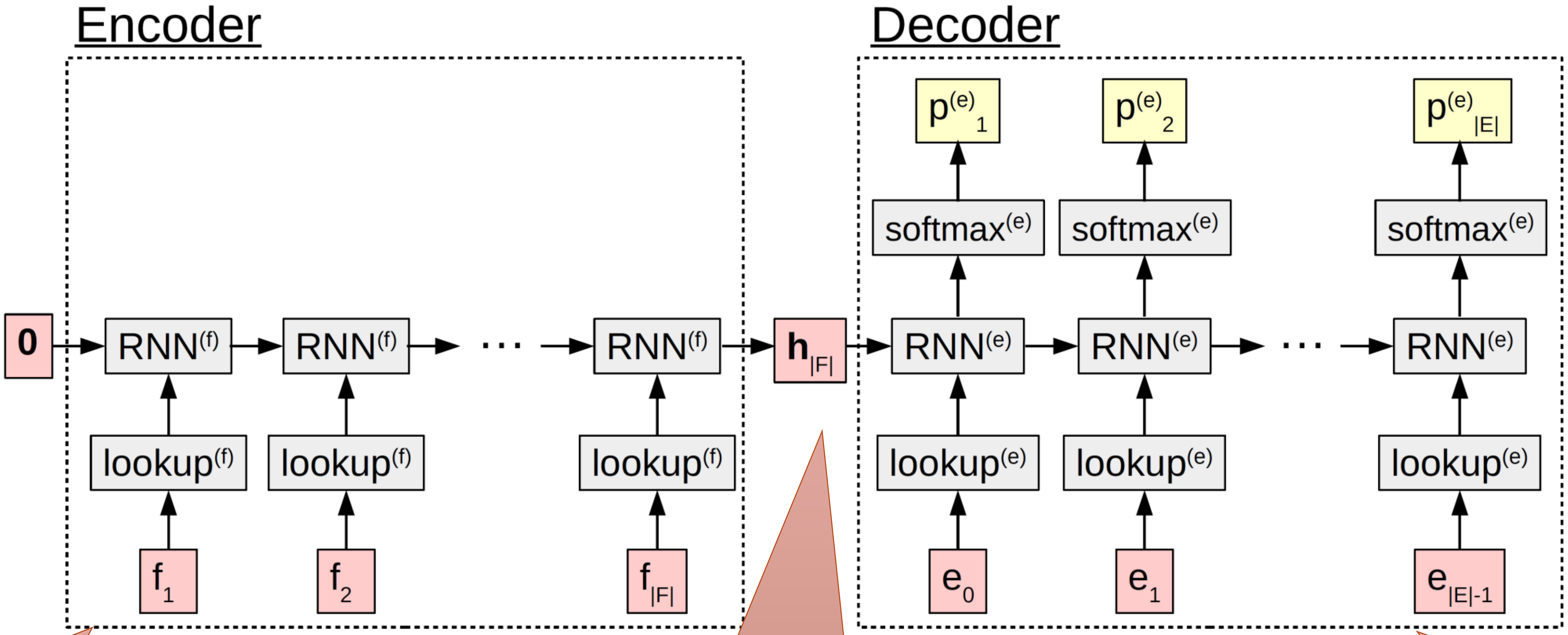
CMSC 470

Marine Carpuat

Introduction to Neural Machine Translation

- Neural language models review
- Sequence to sequence models for MT
  - Encoder-Decoder
  - Sampling and search (greedy vs beam search)
  - **Practical tricks**
- **Attention mechanism**
- **Sequence to sequence models for other NLP tasks**

P(E|F) as an encoder-decoder model



The Encoder models the input/source entence F=(f1,….f|F|)

The decoder hidden state is initialized with the last hidden state of the encoder

The Decoder models the output/target sentence E=(e1,….e|E|).

P(E|F) as an encoder-decoder model

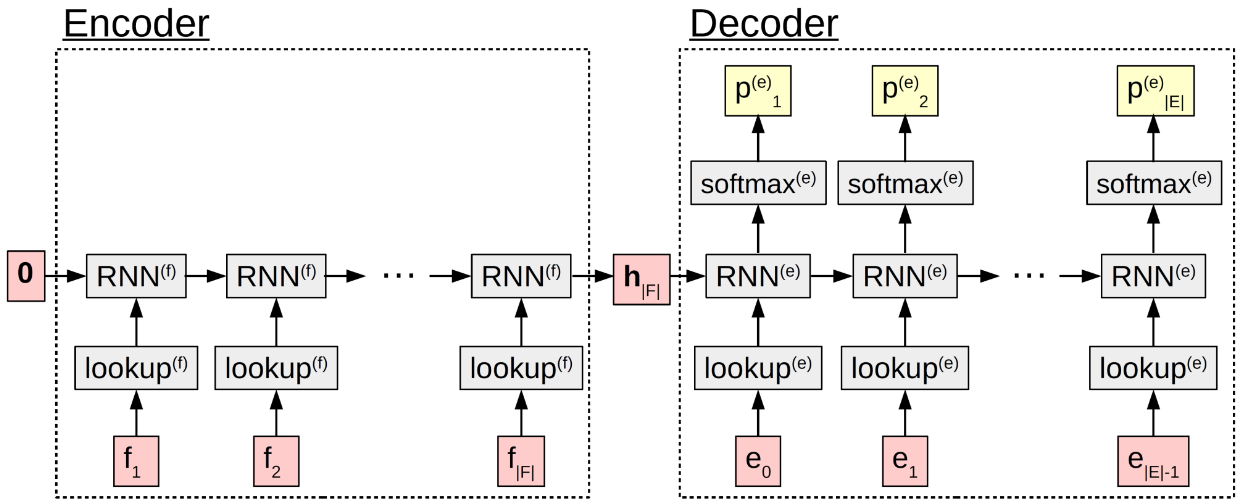
$$\boldsymbol{m}_t^{(f)} = M^{(f)}_{\cdot, f_t}$$

$$\boldsymbol{h}_t^{(f)} = \begin{cases} \mathrm{RNN}^{(f)}(\boldsymbol{m}_t^{(f)}, \boldsymbol{h}_{t-1}^{(f)}) & t \geq 1, \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

$$\boldsymbol{m}_t^{(e)} = M^{(e)}_{\cdot, e_{t-1}}$$

$$\boldsymbol{h}_t^{(e)} = \begin{cases} \mathrm{RNN}^{(e)}(\boldsymbol{m}_t^{(e)}, \boldsymbol{h}_{t-1}^{(e)}) & t \geq 1, \\ \boldsymbol{h}_{|F|}^{(f)} & \text{otherwise.} \end{cases}$$

$$\boldsymbol{p}_t^{(e)} = \mathrm{softmax}(W_{hs}\boldsymbol{h}_t^{(e)} + b_s)$$



Problem with previous encoder-decoder model

- This approach doesn't quite work…
  - Lots of data + lots of tricks needed to get translations that are not horrible

- Why?
  - Long-distance dependencies remain a problem
  - A single vector represents the entire source sentence
    - No matter its length

- The **attention mechanism** helps address this issue
  - An example of incorporating inductive bias in model architecture

Attention model intuition

- Encode each word in source sentence into a vector
- When decoding, perform a linear combination of these vectors, weighted by “attention weights”
- Use this combination when predicting next word

[Bahdanau et al. 2015]

Attention model
Source word representations

- We can use representations from bidirectional RNN encoder

$$\overrightarrow{\boldsymbol{h}}_j^{(f)} = \mathrm{RNN}(\mathrm{embed}(f_j), \overrightarrow{\boldsymbol{h}}_{j-1}^{(f)})$$
$$\overleftarrow{\boldsymbol{h}}_j^{(f)} = \mathrm{RNN}(\mathrm{embed}(f_j), \overleftarrow{\boldsymbol{h}}_{j+1}^{(f)}).$$

$$\boldsymbol{h}_j^{(f)} = [\overleftarrow{\boldsymbol{h}}_j^{(f)}; \overrightarrow{\boldsymbol{h}}_j^{(f)}].$$

- And concatenate them in a matrix

$$H^{(f)} = \mathrm{concat\_col}(\boldsymbol{h}_1^{(f)}, \ldots, \boldsymbol{h}_{|F|}^{(f)}).$$

Attention model: at each decoding time step t, create a source context vector c_t

$$\boldsymbol{c}_t = H^{(f)}\boldsymbol{\alpha}_t.$$

Context vector

Attention vector

- Attention vector α_t:
  - Entries between 0 and 1
  - Interpreted as weight given to each source word when generating output at time step t

- Used to combine source representations into a context vector c_t

Attention model

$$\boldsymbol{h}_t^{(e)} = \text{enc}([\text{embed}(e_{t-1}); \boldsymbol{c}_{t-1}], \boldsymbol{h}_{t-1}^{(e)}).$$

$$a_{t,j} = \text{attn\_score}(\boldsymbol{h}_j^{(f)}, \boldsymbol{h}_t^{(e)}).$$

$$\boldsymbol{\alpha}_t = \text{softmax}(\boldsymbol{a}_t).$$

$$\boldsymbol{p}_t^{(e)} = \text{softmax}(W_{hs}[\boldsymbol{h}_t^{(e)}; \boldsymbol{c}_t] + b_s).$$

The context vector is concatenated with the decoder hidden state to generate the next target word

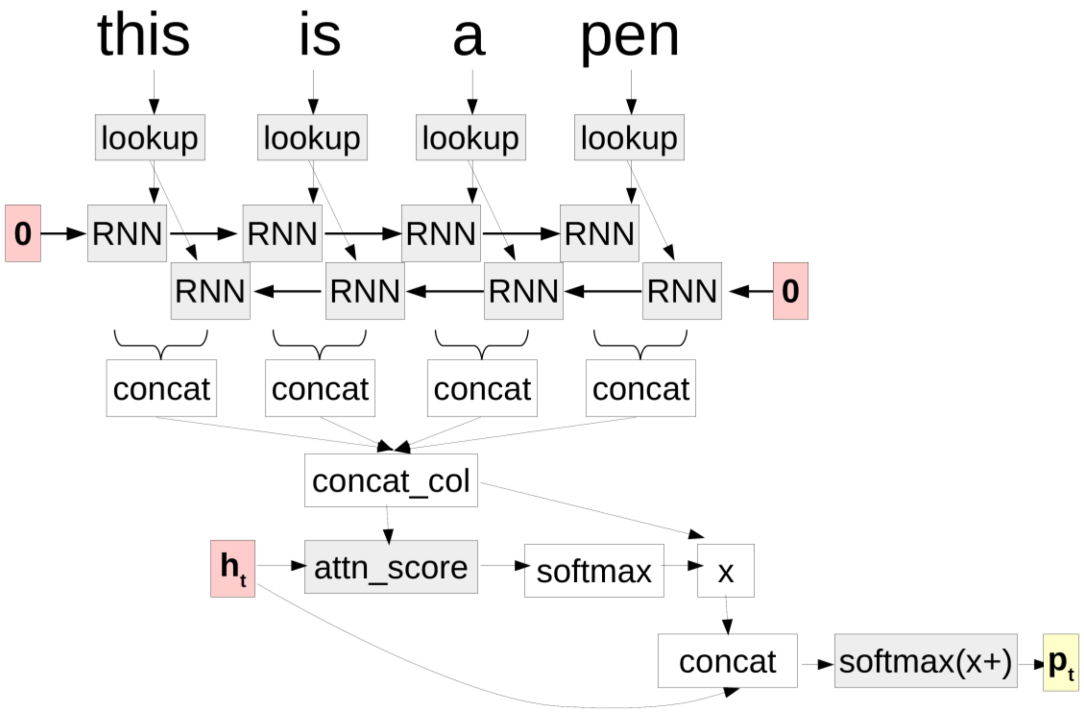


Figure 28: A computation graph for attention.

Attention model
Various ways of calculating attention score

- Dot product

$$\text{attn\_score}(\boldsymbol{h}_j^{(f)}, \boldsymbol{h}_t^{(e)}) := \boldsymbol{h}_j^{(f)\intercal} \boldsymbol{h}_t^{(e)}.$$

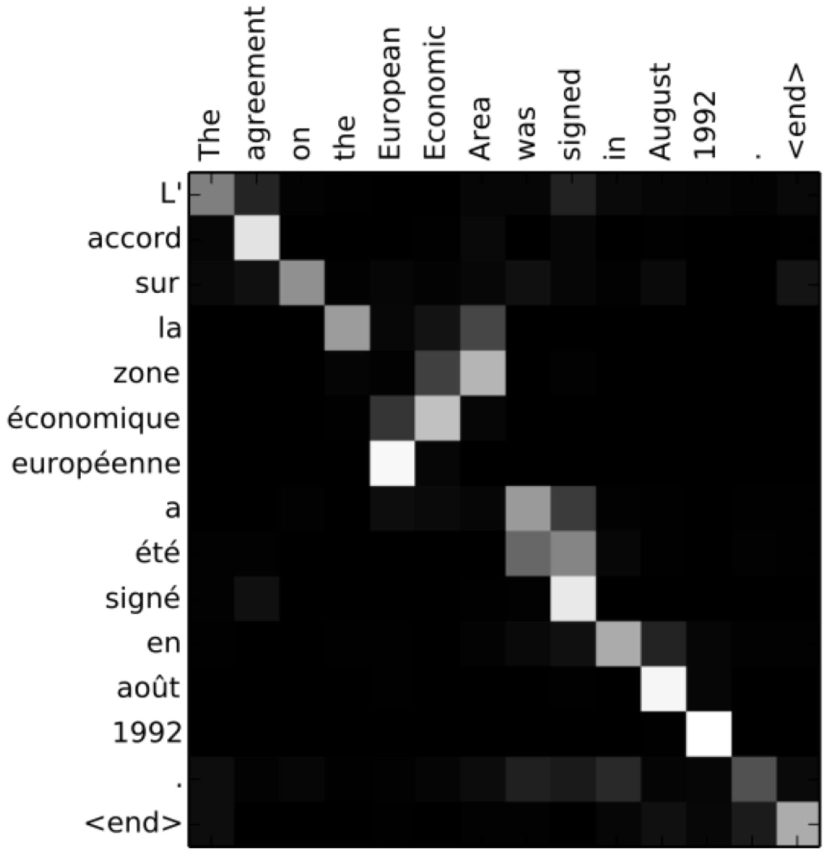
- Bilinear function

$$\text{attn\_score}(\boldsymbol{h}_j^{(f)}, \boldsymbol{h}_t^{(e)}) := \boldsymbol{h}_j^{(f)\intercal} W_a \boldsymbol{h}_t^{(e)}.$$

- Multi-layer perceptron (original formulation in Bahdanau et al.)

$$\text{attn\_score}(\boldsymbol{h}_t^{(e)}, \boldsymbol{h}_j^{(f)}) := \boldsymbol{w}_{a2}^{\intercal} \tanh(W_{a1}[\boldsymbol{h}_t^{(e)}; \boldsymbol{h}_j^{(f)}])$$

Attention model
Illustrating attention weights



Advantages of attention

- Helps illustrate/interpret translation decisions

- Can help insert translations for out-of-vocabulary words
  - By copying or look up in external dictionary

- Can incorporate linguistically motivated priors in model

Attention extensions
Bidirectional constraints (Cohn et al. 2015)

- Intuition: attention should be similar in forward and backward translation directions

- Method: train so that we get a bonus based on the trace of matrix product for training in both directions

$$\operatorname{tr}(A_{X \to Y} A_{Y \to X}^{\intercal})$$

Attention extensions
An active area of research

- Attend to multiple sentences (Zoph et al. 2015)
- Attend to a sentence and an image (Huang et al. 2016)

A few more tricks: addressing length bias

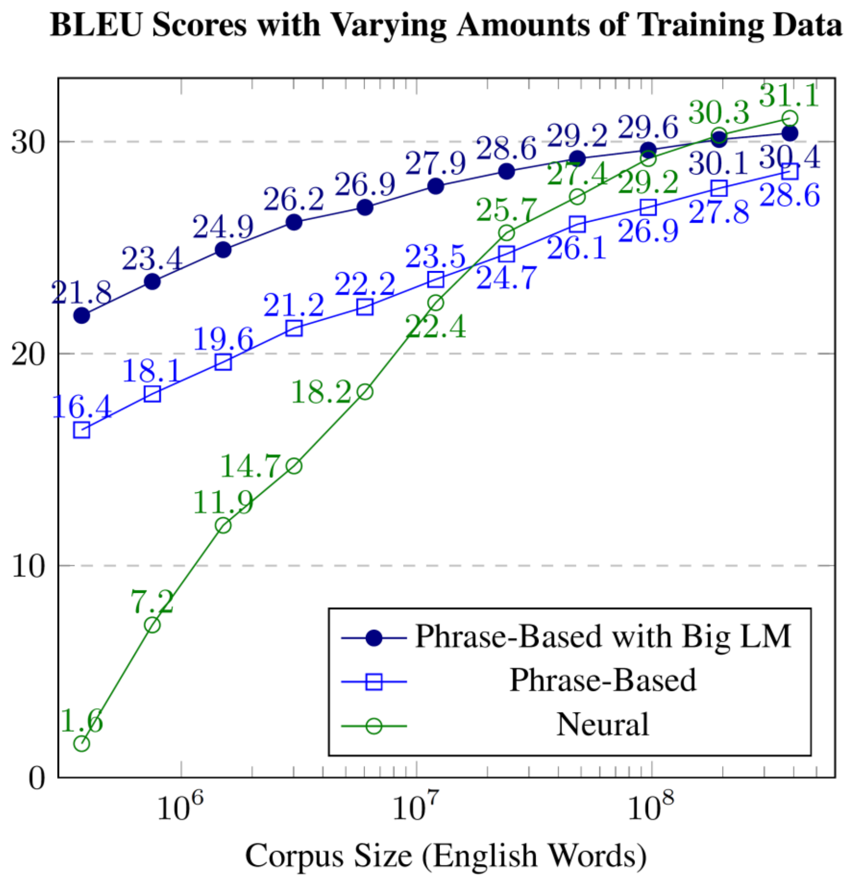
- Default models tend to generate short sentences
- Solutions:
  - Prior probability on sentence length

$$\hat{E} = \underset{E}{\operatorname{argmax}} \ \log P(|E| \mid |F|) + \log P(E \mid F).$$

  - Normalize by sentence length

$$\hat{E} = \underset{E}{\operatorname{argmax}} \ \log P(E \mid F)/|E|.$$

Issue with Neural MT: it only works well in high-resource settings



[Koehn & Knowles 2017]

Ongoing research

- Learn from other sources of supervision than pairs (E,F)
  - Monolingual text
  - Multiple languages
- Incorporate linguistic knowledge
  - As additional embeddings
  - As prior on network structure or parameters
  - To make better use of training data

State-of-the-art neural MT models are very powerful, but still make many errors

https://www.youtube.com/watch?v=3-rfBsWmo0M

Neural Machine Translation
What you should know

- How to formulate machine translation as a sequence-to-sequence transformation task
- How to model P(E|F) using RNN encoder-decoder models, with and without attention
- Algorithms for producing translations
  - Ancestral sampling, greedy search, beam search
- How to train models
  - Computation graph, batch vs. online vs. minibatch training
- Examples of weaknesses of neural MT models and how to address them
  - Bidirectional encoder, length bias
- Determine whether a NLP task can be addressed with neural sequence-to-sequence models