Go to

join.iClicker.com

HXFY

COS 484

Natural Language Processing

L12: Seq2seq models + attention

Spring 2025

Final project guideline

- **Deadline**: March 28th 11:59pm
  - Submission via Gradescope (use group submission; add your team members!)

- **Options:**
  - (a) Reproducing a paper - highly encouraged

    A recent ACL/NAACL/EMNLP/COLM paper; NLP papers in NeurIPS/ICLR/ICML are fine too!

    We have curated a list for paper suggestions
  - (b) Complete a research project - we ask you to get approval by making a private Ed post

- We have posted project guidelines on the website

Final project guideline

- **How to do NLP research in 2025?**
  - Route #1: call LLM APIs (e.g., OpenAI, Claude)
  - Route #2: download an open-weight model and run/fine-tune it on GPUs (e.g., Llama-3, Gemma-3 models)
- Please think carefully about your needs and make it as concrete as possible in your proposal
  - Some models are cheaper to call
  - Small language models are incredibly powerful (e.g., llama 3.2, gemma3 1b models)
- We will reimburse each team Colab Pro (2 months) and/or API credits up to a small budget
- We encourage you to explore other computing or free LLM API resources

Neural machine translation (NMT)

- Neural Machine Translation (NMT) is a way to do machine translation with a **single end-to-end neural network**

Sequence to Sequence Learning with Neural Networks

Ilya Sutskever
Google
ilyasu@google.com

Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
qvl@google.com

(Sutskever et al., 2014)

Ilya Sutskever

- The neural network architecture is called a **sequence-to-sequence model** (aka **seq2seq**) and it involves two RNNs

Google's NMT system in 2016

RESEARCH › PUBLICATIONS ›

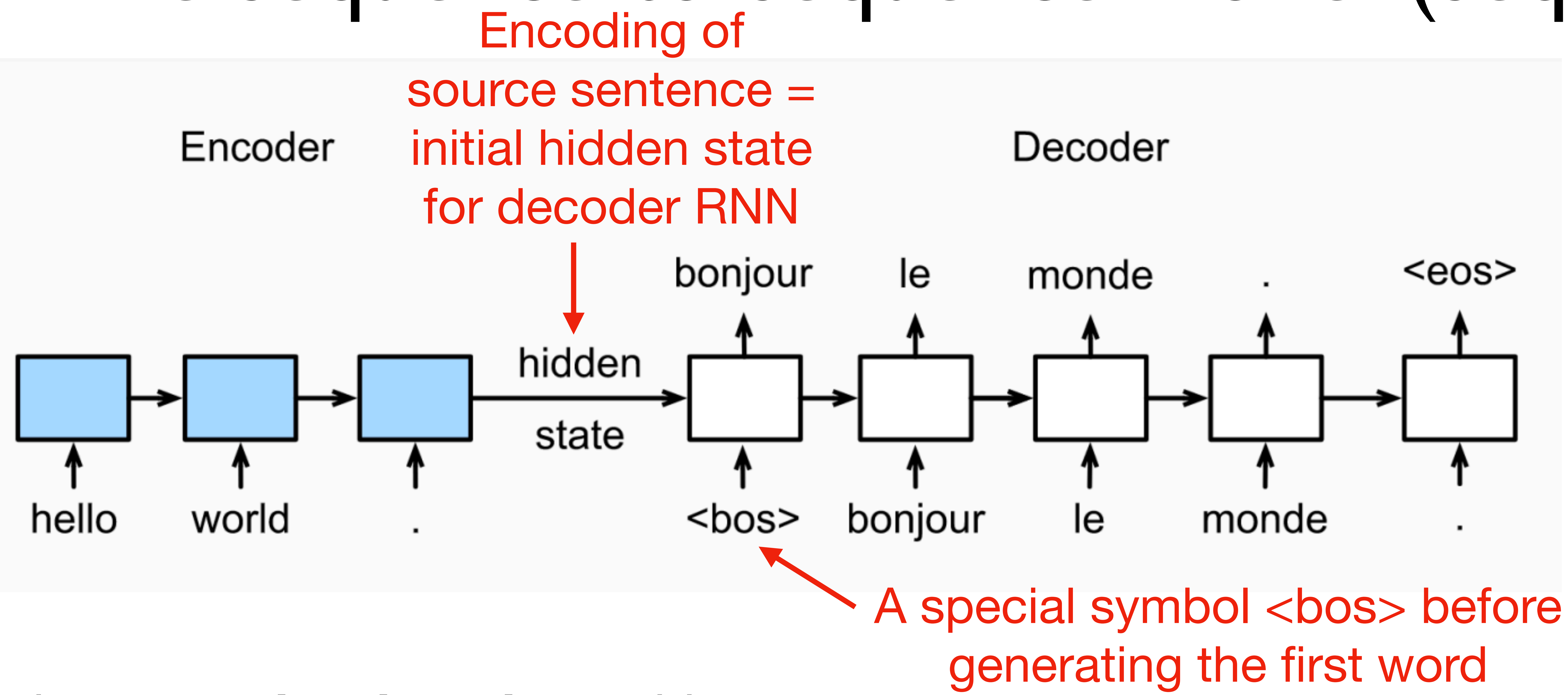
Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

Table 10: Mean of side-by-side scores on production data

	PBMT	GNMT	Human	Relative Improvement
English → Spanish	4.885	5.428	5.504	87%
English → French	4.932	5.295	5.496	64%
English → Chinese	4.035	4.594	4.987	58%
Spanish → English	4.872	5.187	5.372	63%
French → English	5.046	5.343	5.404	83%
Chinese → English	3.694	4.263	4.636	60%

(Wu et al., 2016): Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

The sequence-to-sequence model (seq2seq)



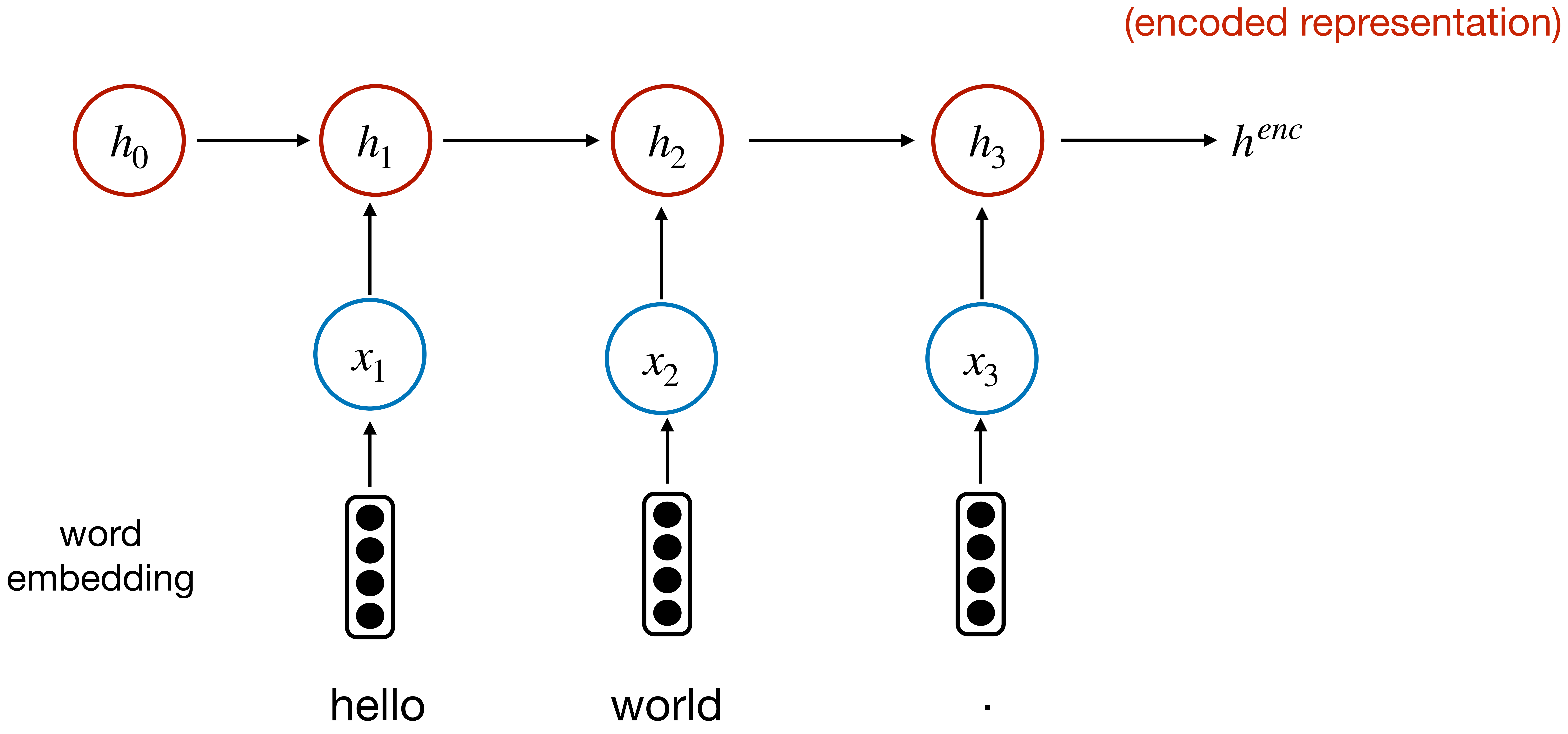
It is called an **encoder-decoder** architecture

- The encoder is an RNN to read the input sequence (source language)
- The decoder is another RNN to generate output word by word (target language)

Image: https://d2l.ai/chapter_recurrent-modern/seq2seq.html

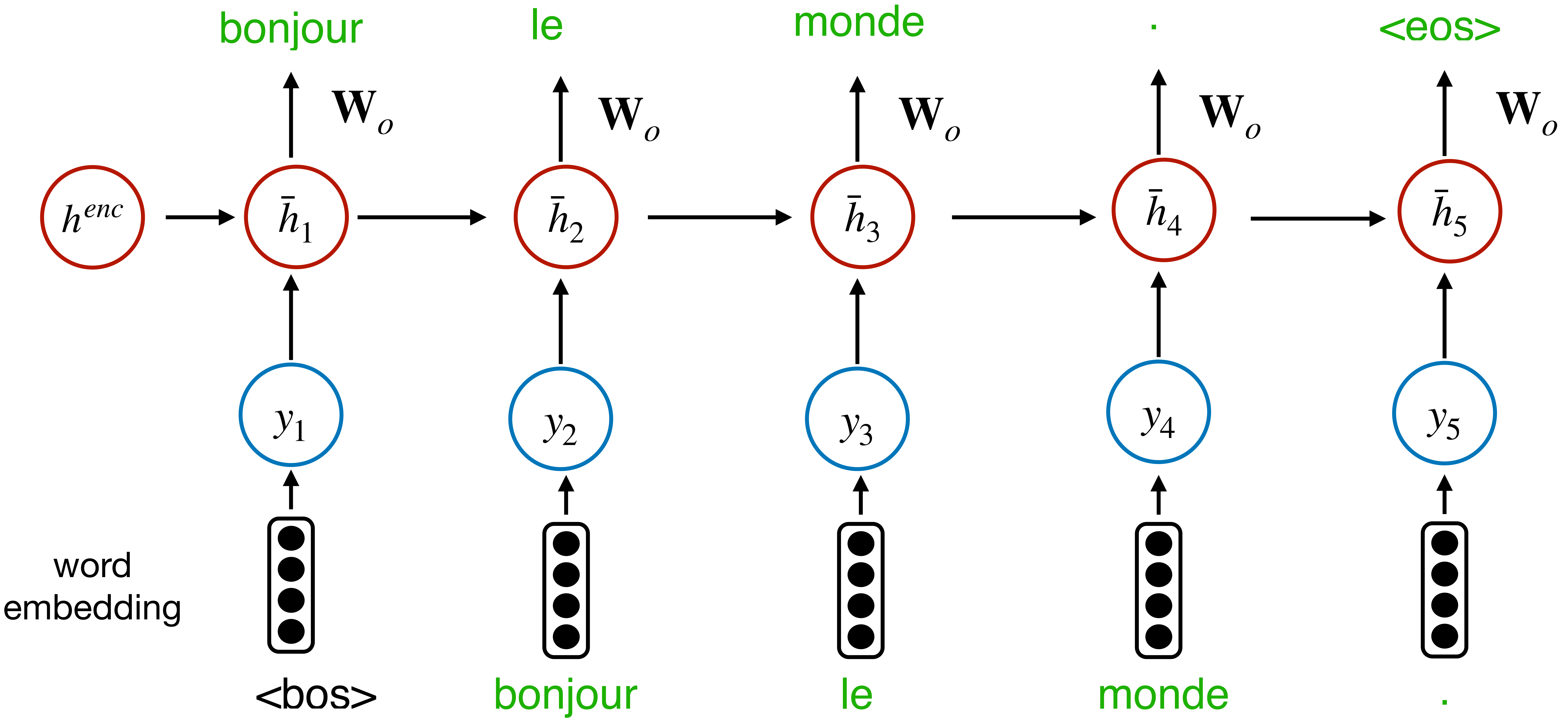
Seq2seq: Encoder

Sentence: hello world .

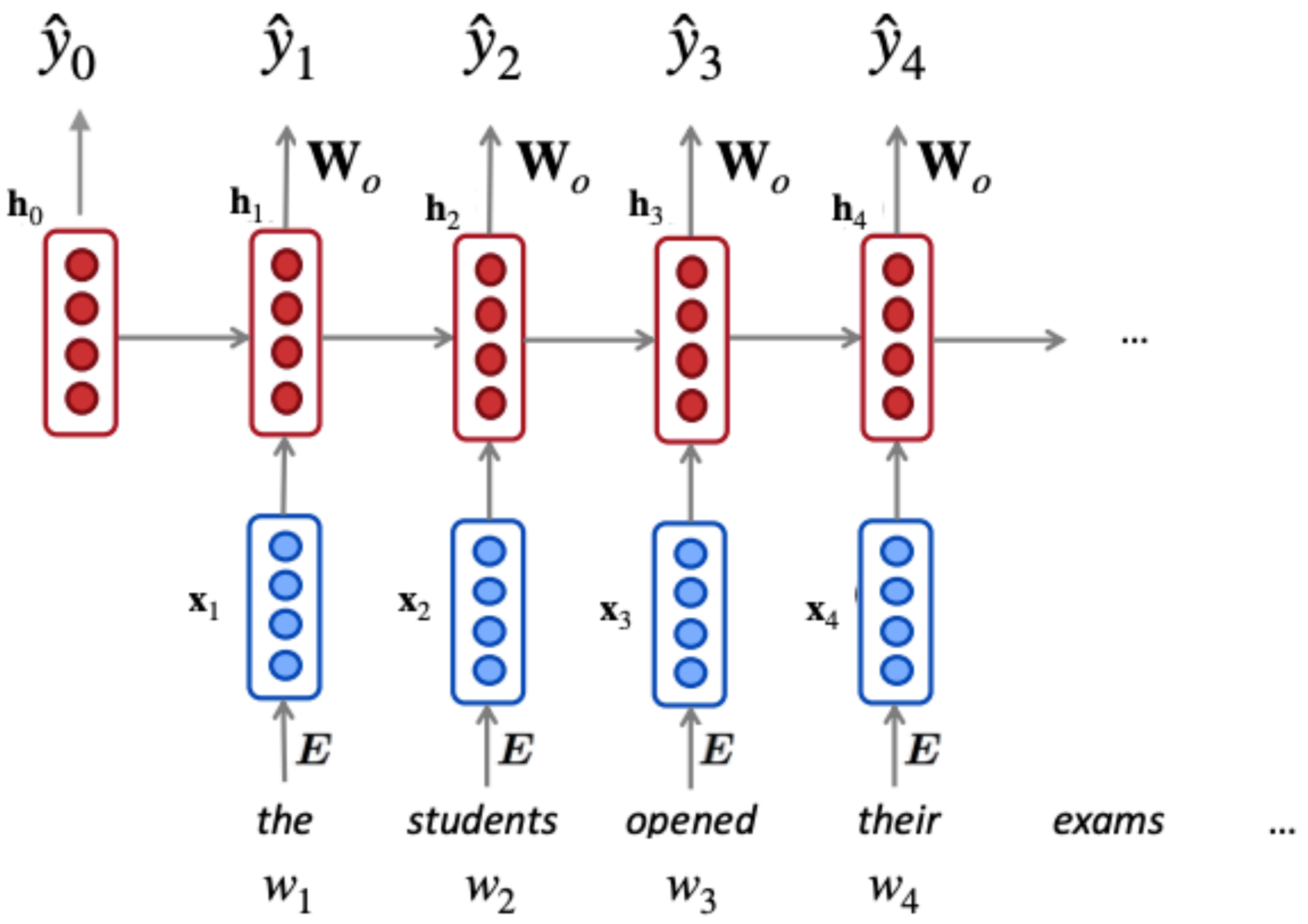


Seq2seq: Decoder

- A **conditional** language model



Recap: recurrent neural language models



$$\mathbf{h}_t = g(\mathbf{W}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{x}_t + \mathbf{b}) \in \mathbb{R}^h$$

$$\hat{\mathbf{y}}_t = softmax(\mathbf{W}_o\mathbf{h}_t)$$

Training loss:

$$\boxed{L(\theta) = -\frac{1}{n}\sum_{t=1}^{n} \log \hat{\mathbf{y}}_{t-1}(w_t)}$$

Trainable parameters:

$$\theta = \{\mathbf{W}, \mathbf{U}, \mathbf{b}, \mathbf{W}_o, \mathbf{E}\}$$

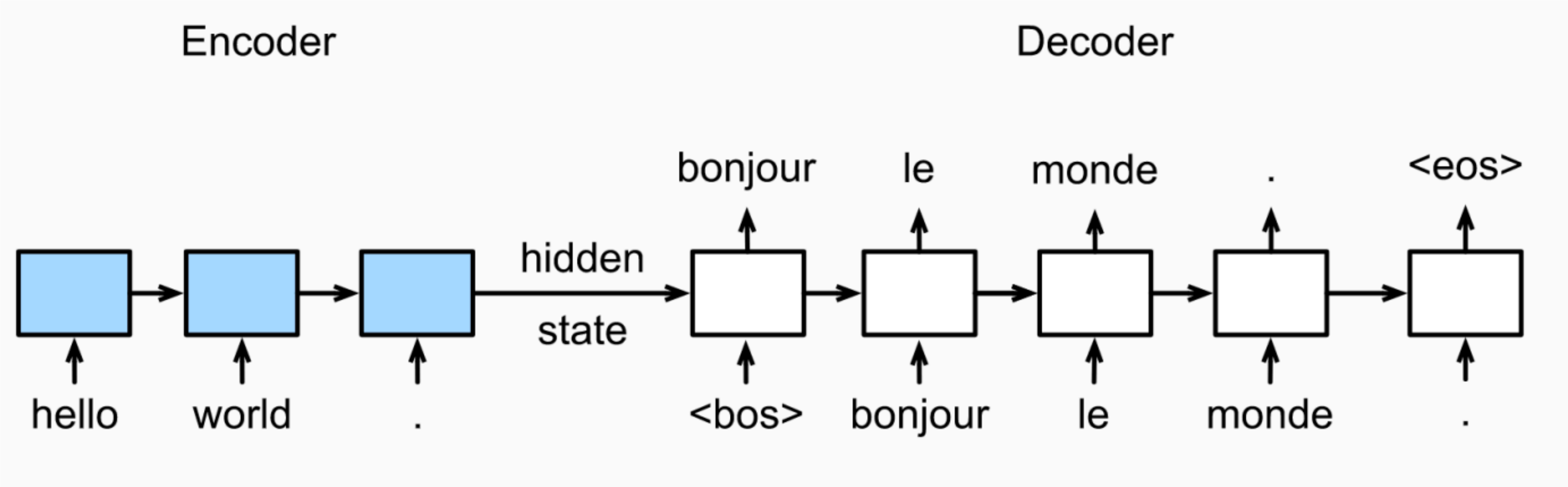
Seq2seq: Decoder

- A **conditional** language model
  - It is a **language model** because the decoder is predicting the next word of the target sentence
  - **Conditional** because the predictions are also conditioned on the source sentence through h^{enc}
- NMT directly calculates $P(\mathbf{w}^{(t)} \mid \mathbf{w}^{(s)})$
  - Denote $\mathbf{w}^{(t)} = y_1, \ldots, y_T$

$$P(\mathbf{w}^{(t)} \mid \mathbf{w}^{(s)}) = P(y_1 \mid \mathbf{w}^{(s)})P(y_2 \mid y_1, \mathbf{w}^{(s)})P(y_3 \mid y_1, y_2, \mathbf{w}^{(s)})\ldots P(y_T \mid y_1, \ldots, y_{T-1}, \mathbf{w}^{(s)})$$

$$\hat{\mathbf{y}}_t = softmax(\mathbf{W}_o\mathbf{h}_t) \quad P(y_{t+1} \mid y_1, \ldots, y_t, \mathbf{w}^{(s)}) = \hat{\mathbf{y}}_t(y_{t+1})$$

Understanding seq2seq

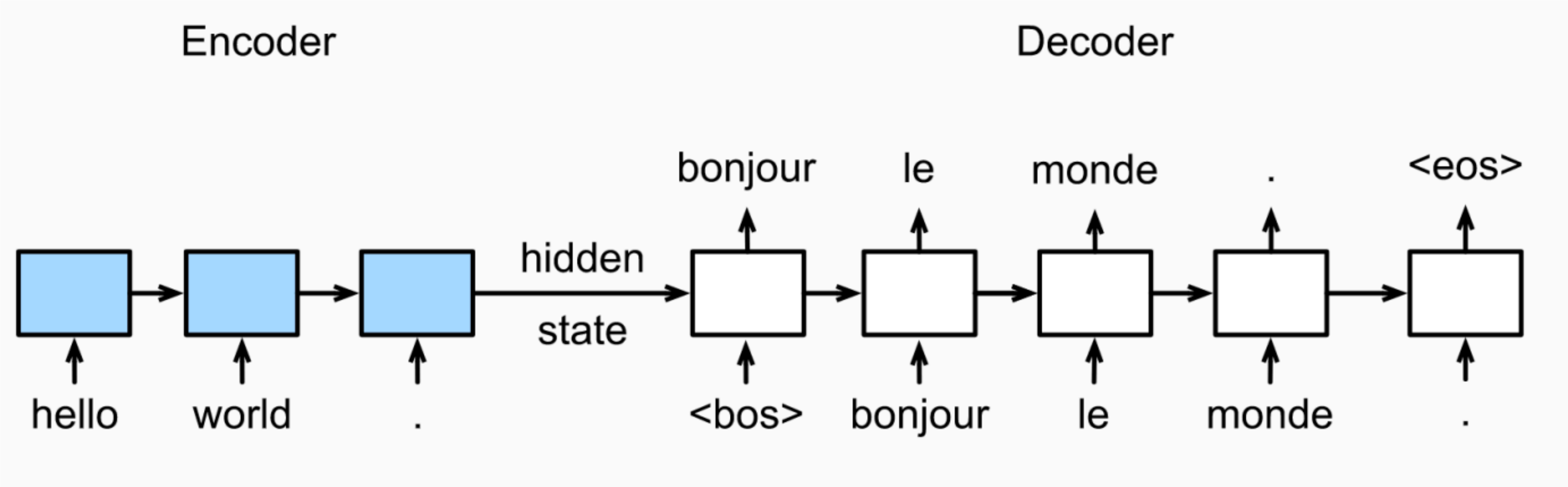


Which of the following is correct?

- (A) We can use bidirectional RNNs for both encoder and decoder
- (B) The decoder has more parameters because of the output matrix $\mathbf{W}_o$
- (C) The encoder and decoder have separate word embeddings
- (D) The encoder and decoder's parameters are optimized together

Both (C) and (D) are correct.

Understanding seq2seq



Encoder RNN:

- word embeddings $\mathbf{E}^{(s)}$ for source language
- RNN parameters, e.g., $\{\mathbf{W}, \mathbf{U}, \mathbf{b}\}$ for simple RNNs and 4x parameters for LSTMs
- Encoder RNN can be bidirectional!

Decoder RNN:

- word embeddings $\mathbf{E}^{(t)}$ for target language
- RNN parameters, e.g., $\{\mathbf{W}, \mathbf{U}, \mathbf{b}\}$ for simple RNNs and 4x parameters for LSTMs
- Output embedding matrix $\mathbf{W}_o$ = can be tied with $\mathbf{E}^{(t)}$
- Decoder RNN has to be unidirectional (left to right)!

Training seq2seq models

- Training data: parallel corpus $\{(\mathbf{w}_i^{(s)}, \mathbf{w}_i^{(t)})\}$

- Minimize cross-entropy loss:

$$\sum_{t=1}^{T} -\log P(y_t \mid y_1, \ldots, y_{t-1}, \mathbf{w}^{(s)})$$

(denote $\mathbf{w}^{(t)} = y_1, \ldots, y_T$)

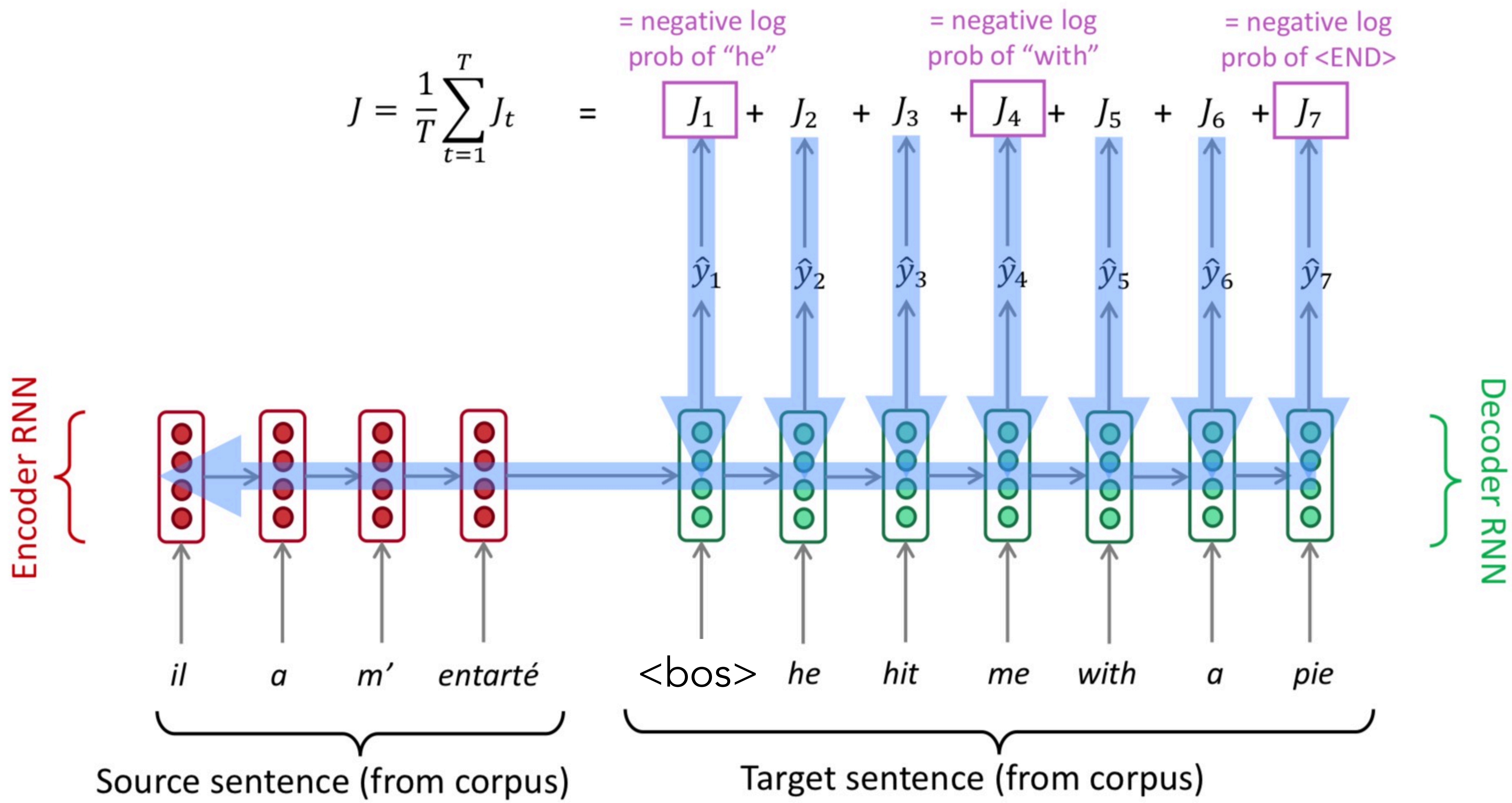
- Back-propagate gradients through both encoder and decoder

12M sentence pairs

French: bonjour le monde .

English: hello world .

Training seq2seq models

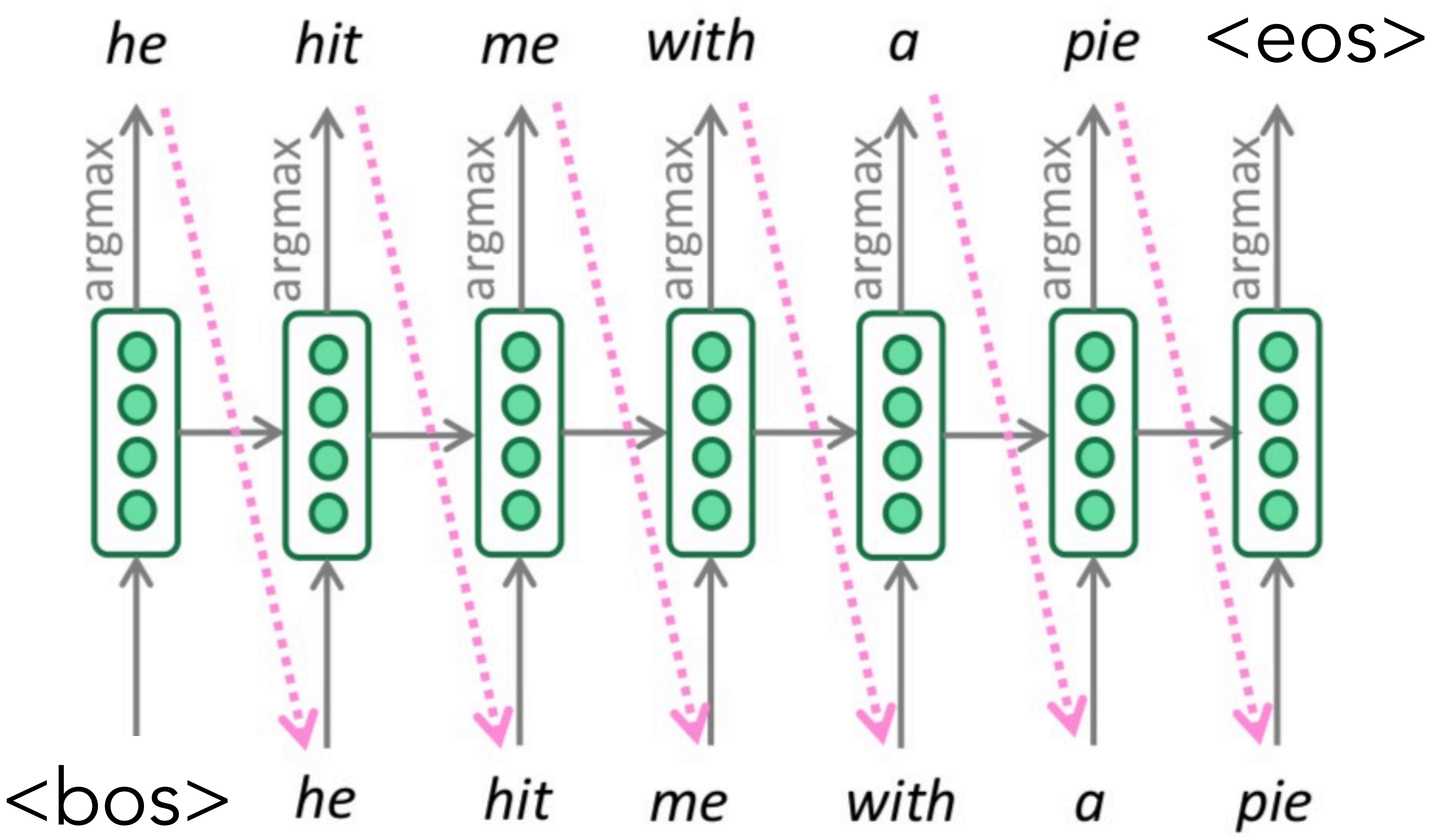


Seq2seq is optimized as a **single system.**
Backpropagation operates *"end-to-end"*.

Decoding seq2seq models

- Greedy decoding

  = Compute argmax at every step of decoder to generate word



- Exhaustive search is very expensive: $\arg\max_{y_1,\ldots,y_T} P(y_1,\ldots,y_T \mid \mathbf{w}^{(s)})$ - we even don't know what T is

A middle ground: Beam search

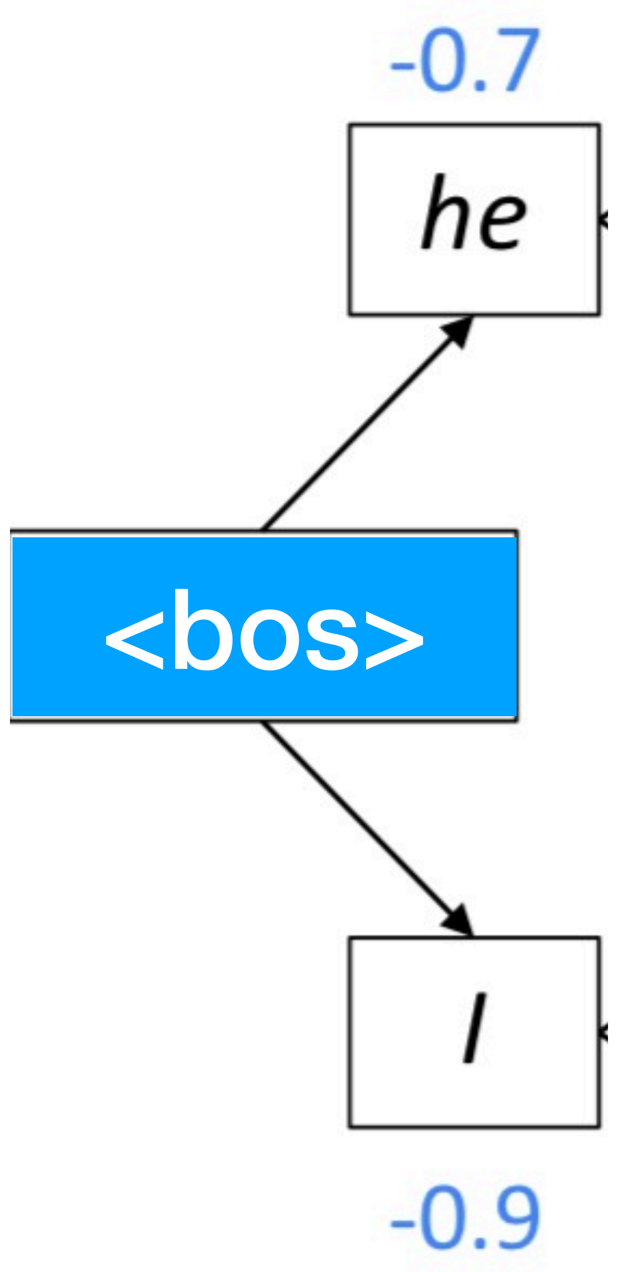
- At every step, keep track of the k most probable partial translations (hypotheses)
- Score of each hypothesis = log probability of sequence so far

$$\sum_{i=1}^{t} \log P(y_i \,|\, y_1, \ldots, y_{i-1}, \mathbf{w}^{(s)})$$

- Not guaranteed to be optimal
- More efficient than exhaustive search

Beam search

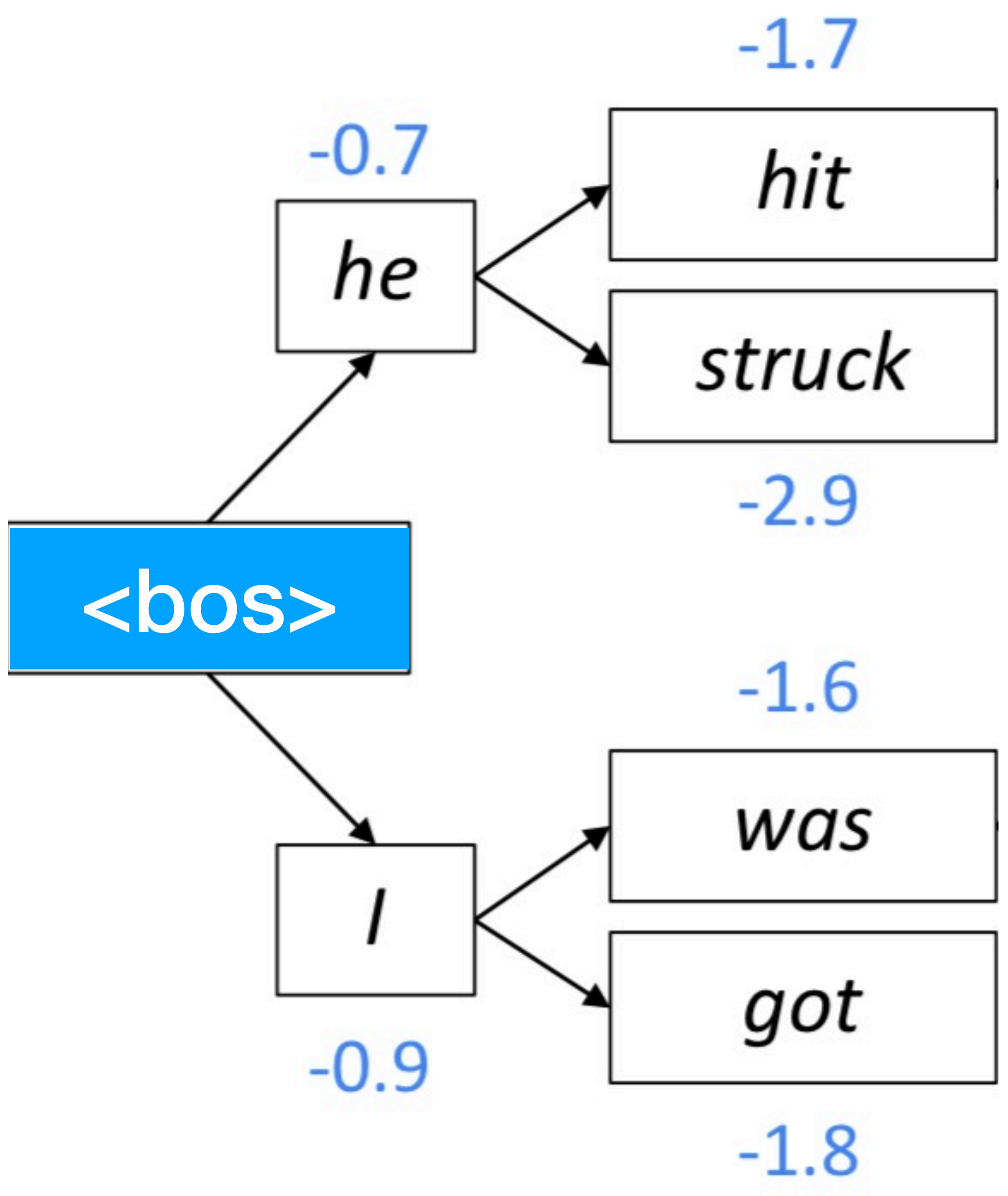
Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



(slide credit: Abigail See)

Beam search

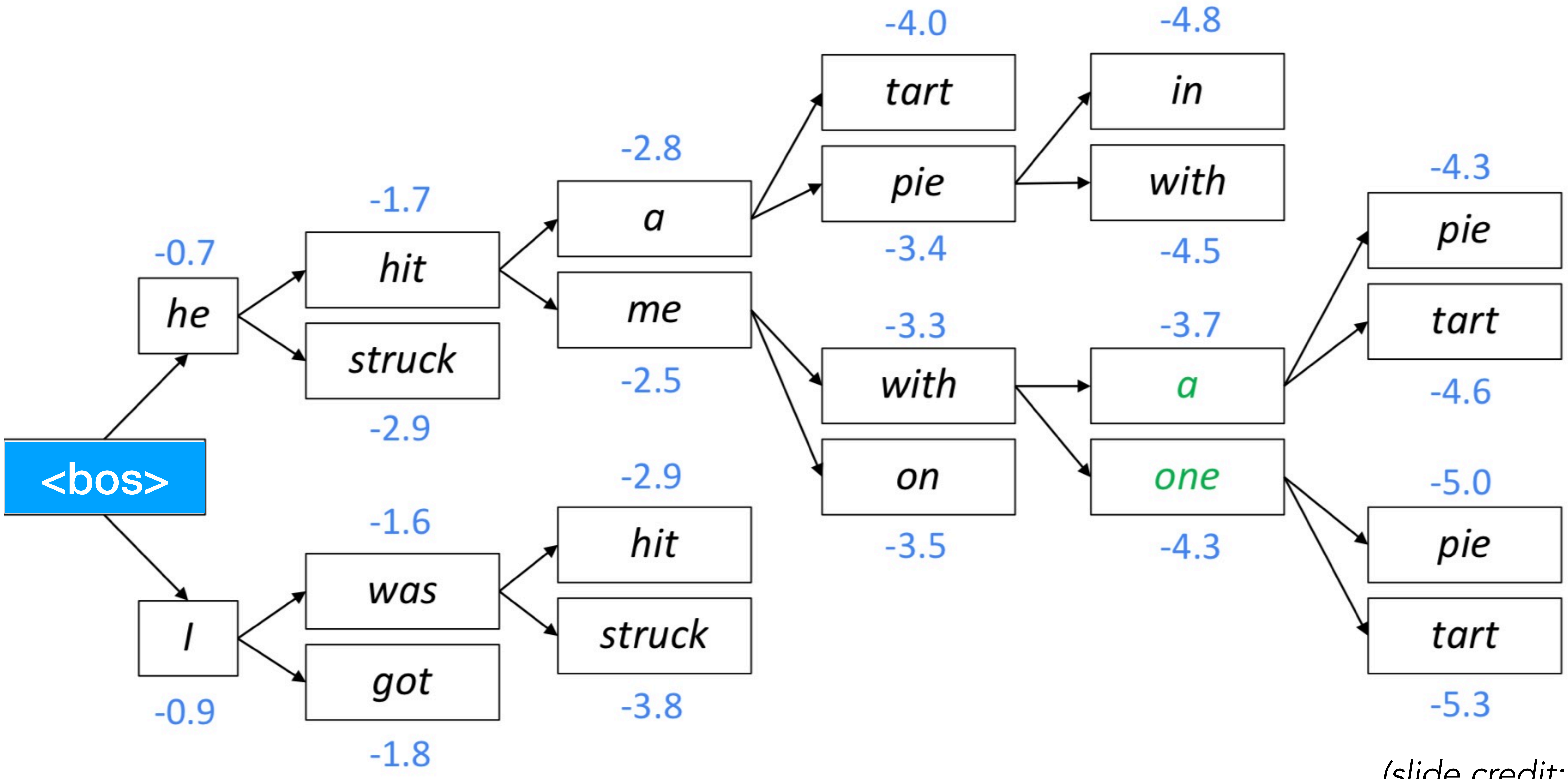
Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



(slide credit: Abigail See)

Beam search

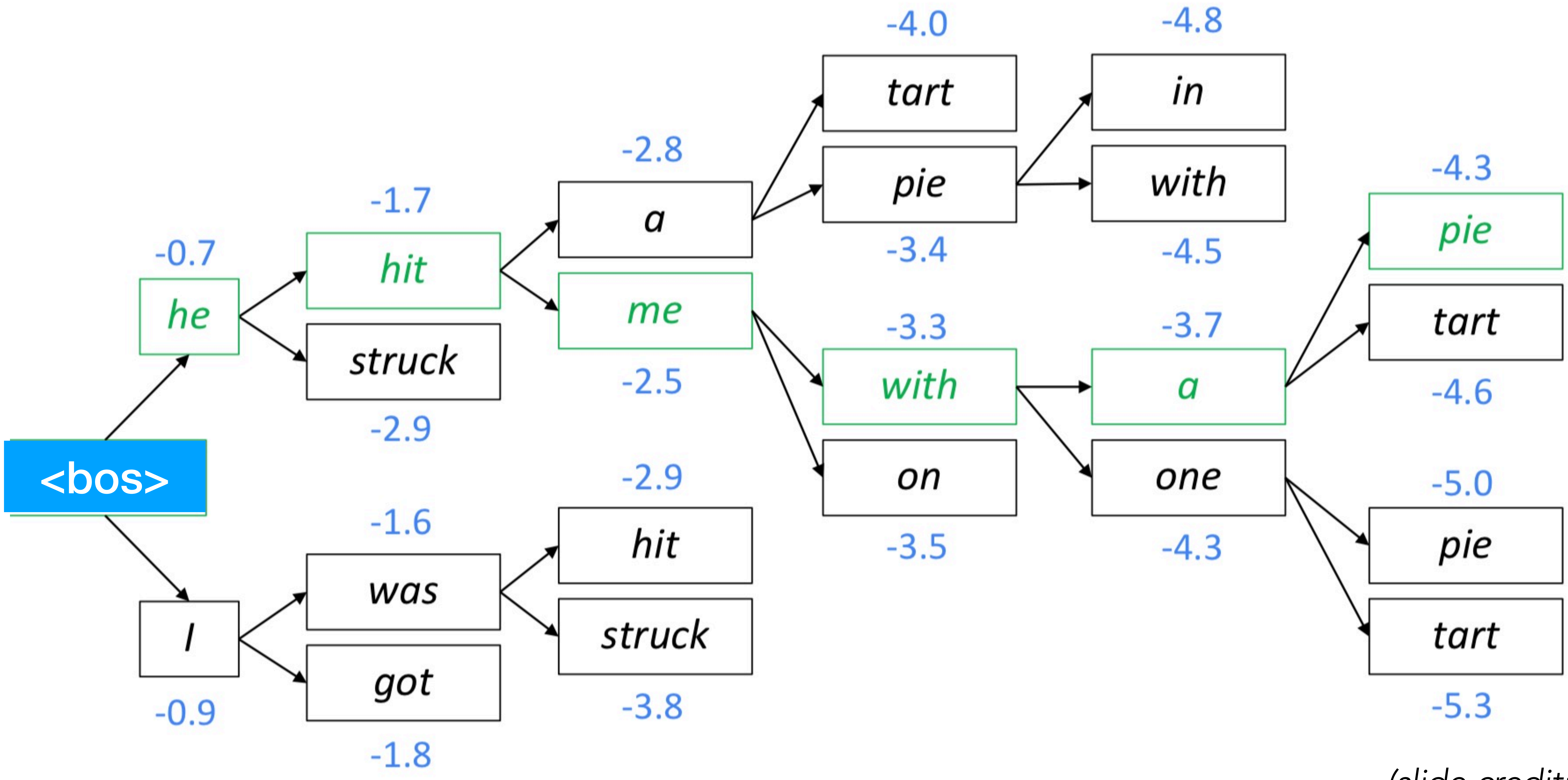
Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



(slide credit: Abigail See)

Backtrack

Beam size = k = 2. Blue numbers = $\text{score}(y_1,\ldots,y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i|y_1,\ldots,y_{i-1},x)$



(slide credit: Abigail See)

Beam search: details

- Different hypotheses may produce $\langle eos \rangle$ token at different time steps
  - When a hypothesis produces $\langle eos \rangle$, stop expanding it and place it aside
- Continue beam search until:
  - All k hypotheses produce $\langle eos \rangle$ OR
  - Hit max decoding limit T
- Select top hypotheses using the *normalized* likelihood score

$$\frac{1}{T}\sum_{t=1}^{T} \log P(y_t \,|\, y_1, \ldots, y_{t-1}, \mathbf{w}^{(s)})$$

  - Otherwise shorter hypotheses have higher scores

NMT vs SMT

Pros:

- Better performance (more **fluent**, better use of **context**, better use of **phrase similarities**)
- A **single neural network** to be optimized end-to-end (no individual subcomponents)
- **Less human engineering effort** - same method for all language pairs

Cons:

- NMT is **less interpretable**
- NMT is **difficult to control**

NMT: the first big success story of NLP deep learning

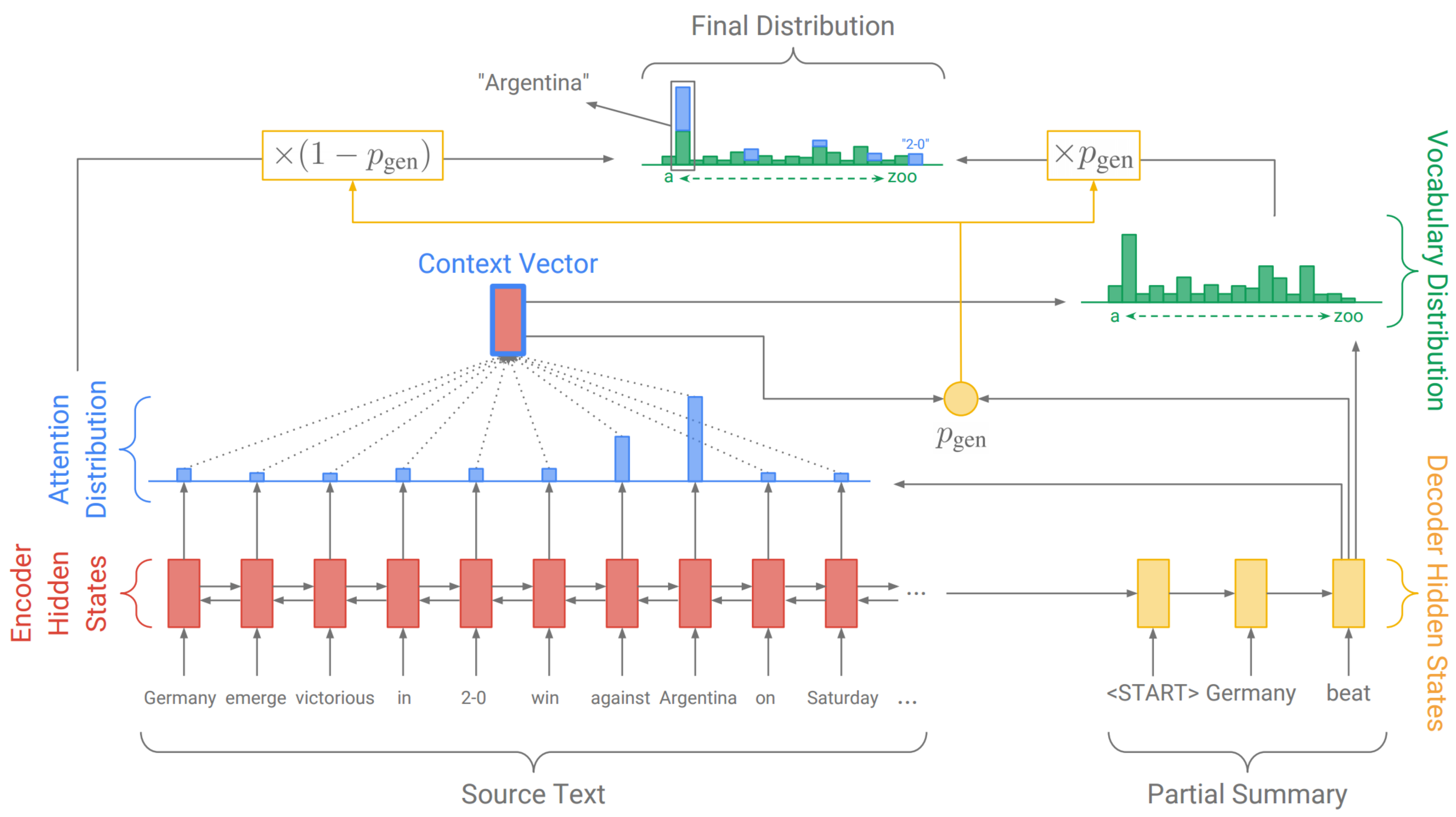
- 2014: First seq2seq paper published
- 2016: Google Translate switches from SMT to NMT - and by 2018 everyone has
- SMT systems, built by hundreds of engineers over many years, outperformed by NMT systems trained by a small group of engineers in a few months

Sequence-to-sequence is versatile

- Sequence-to-sequence is useful for more than just MT
- Many NLP tasks can be framed as sequence-to-sequence problems
    - **Summarization** (long text → short text)
    - **Dialogue** (previous utterances → next utterance)
    - **Code generation** (natural language → Python code)
    - …

Sequence-to-sequence is versatile

- Summarization



Source Text

munster have signed new zealand international francis *saili* on a two-year deal . utility back *saili* , who made his all blacks debut against argentina in 2013 , will move to the province later this year after the completion of his 2015 contractual commitments . the 24-year-old currently plays for *auckland-based* super rugby side the blues and was part of the new zealand under-20 side that won the junior world championship in italy in 2011 . *saili* 's signature is something of a coup for munster and head coach anthony foley believes he will be a great addition to their backline . francis *saili* has signed a two-year deal to join munster and will link up with them later this year . ' we are really pleased that francis has committed his future to the province , ' foley told munster 's official website . ' he is a talented centre with an impressive *skill-set* and he possesses the physical attributes to excel in the northern hemisphere . ' i believe he will be a great addition to our backline and we look forward to welcoming him to munster . ' *saili* has been capped twice by new zealand and was part of the under 20 side that won the junior championship in 2011 . *saili* , who joins all black team-mates dan carter , *ma'a nonu* , conrad smith and charles *piutau* in agreeing to ply his trade in the northern hemisphere , is looking forward to a fresh challenge . he said : ' i believe this is a fantastic opportunity for me and i am fortunate to move to a club held in such high regard , with values and traditions i can relate to from my time here in the blues . ' this experience will stand to me as a player and i believe i can continue to improve and grow within the munster set-up . ' as difficult as it is to leave the blues i look forward to the exciting challenge ahead . '

Reference summary

utility back francis *saili* will join up with munster later this year .
the new zealand international has signed a two-year contract .
saili made his debut for the all blacks against argentina in 2013 .

Sequence-to-sequence + attention summary

dutch international francis [UNK] has signed a two-year deal to join irish [UNK] super rugby side the blues .
[UNK] 's signature is something of a coup for munster and his head coach anthony foley believes he will be a great addition to their respective prospects .
[UNK] has been capped twice by new zealand .

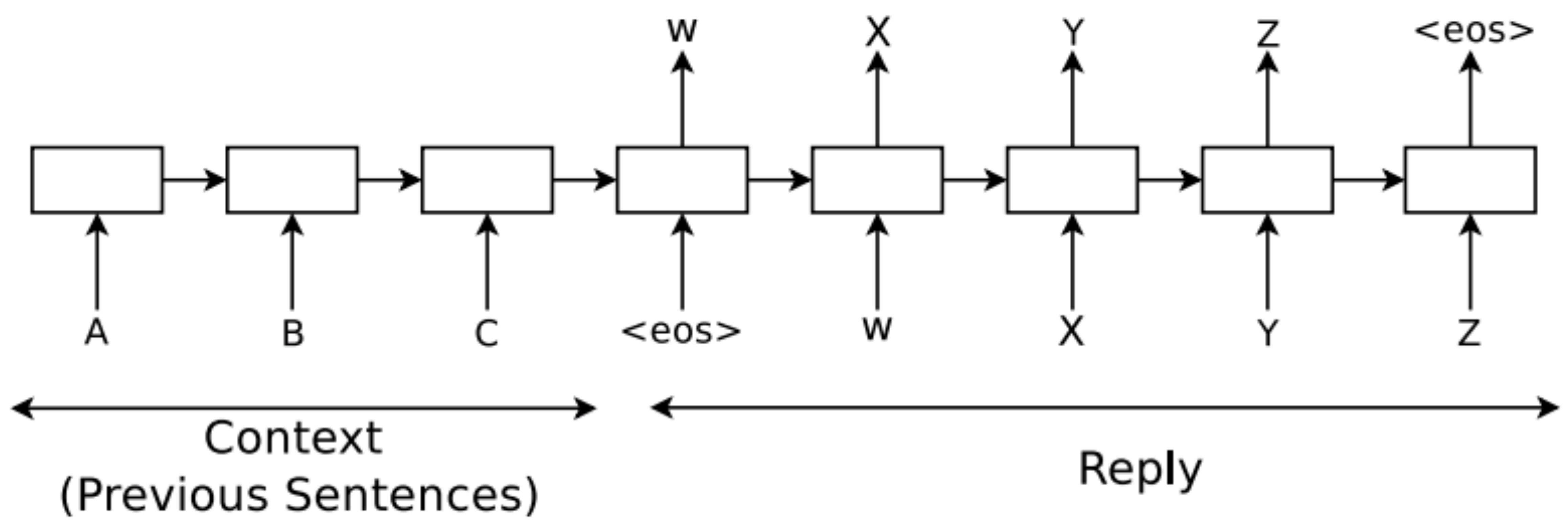
Pointer-generator summary

new zealand international francis *saili* will move to the province later this year .
utility back *saili* made his all blacks debut against argentina in 2013 .
utility back *saili* will move to the province later this year .

See et al., 2017: Get To The Point: Summarization with Pointer-Generator Networks

Sequence-to-sequence is versatile

- Dialogue



Human: *hello !*
Machine: *hello !*
Human: *how are you ?*
Machine: *i 'm good .*
Human: *what 's your name ?*
Machine: *i 'm julia .*
Human: *when were you born ?*
Machine: *july 20th .*
Human: *what year were you born ?*
Machine: *1977 .*
Human: *where are you from ?*

Vinyals and Le 2015: A Neuarl Conversational Model

Sequence-to-sequence is versatile

- Code generation

Problem

H-Index

Given a list of citations counts, where each citation is a nonnegative integer, write a function h_index that outputs the h-index. The h-index is the largest number *h* such that *h* papers have each least *h* citations.

Example:
Input: [3,0,6,1,4]
Output: 3

Generated Code

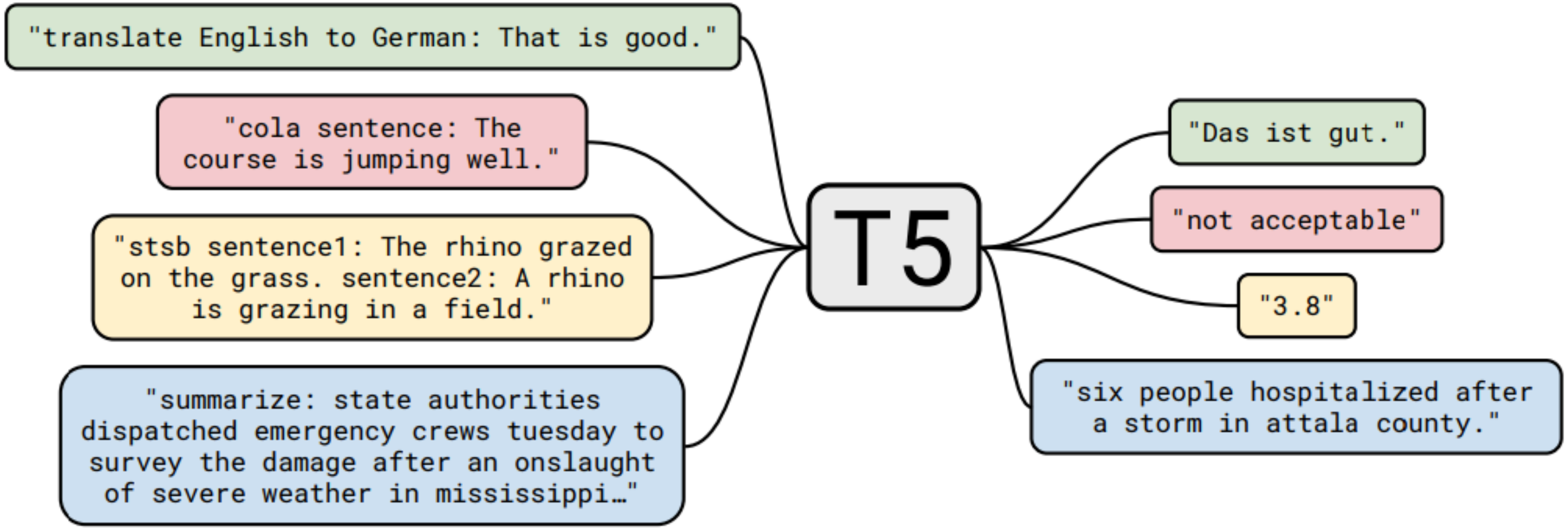
```
def h_index(counts):
  n = len(counts)
  if n > 0:
    counts.sort()
    counts.reverse()
    h = 0
    while (h < n and
        counts[h]-1>=h):
      h += 1
    return h
  else:
    return 0
```

Test Cases

```
Input:
[1,4,1,4,2,1,3,5,6]

Generated Code Output:
4 ✓

Input:
[1000,500,500,250,100,
100,100,100,100,75,50,
30,20,15,15,10,5,2,1]

Generated Code Output:
15 ✓
```

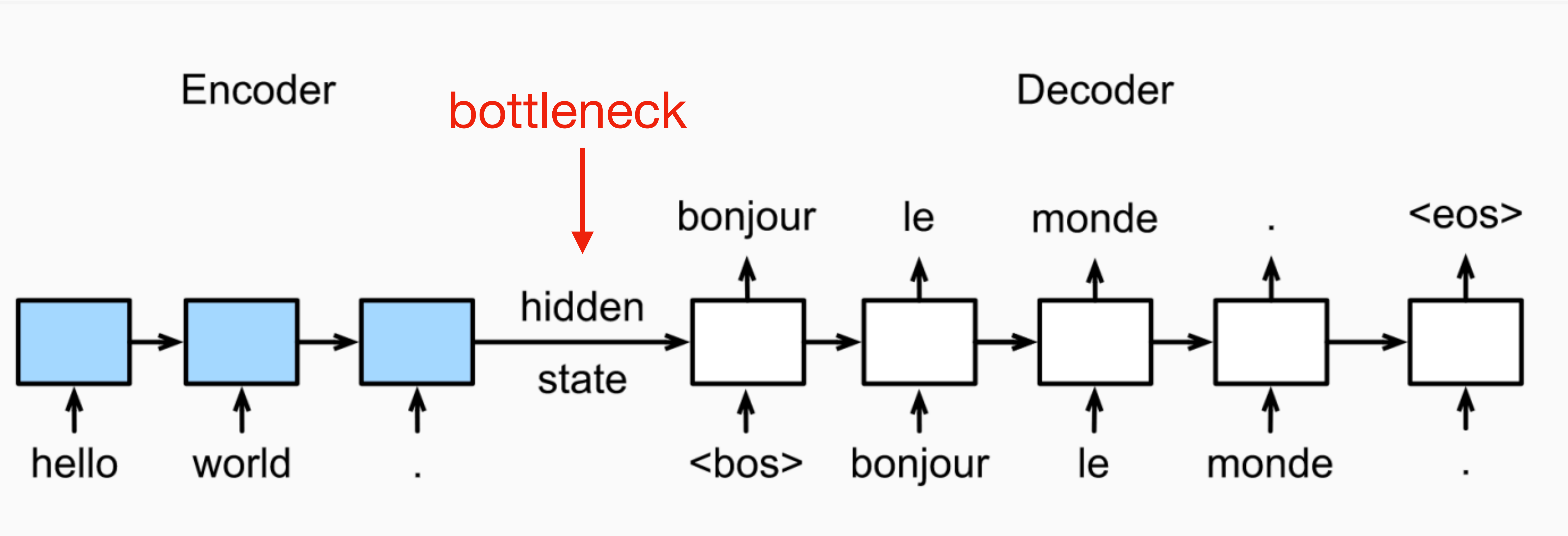
Sequence-to-sequence is versatile

- All language tasks can be converted into a text-to-text problem!
  - T5 = **T**ext-**t**o-**t**ext **T**rasnfer **T**ransformer



Raffel et al., 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Sequence-to-sequence: the bottleneck



- A single encoding vector, h^{enc}, needs to capture **all the information** about source sentence
- Longer sequences can lead to vanishing gradients

Attention

- Attention provides a solution to the bottleneck problem

ICLR 2015

NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

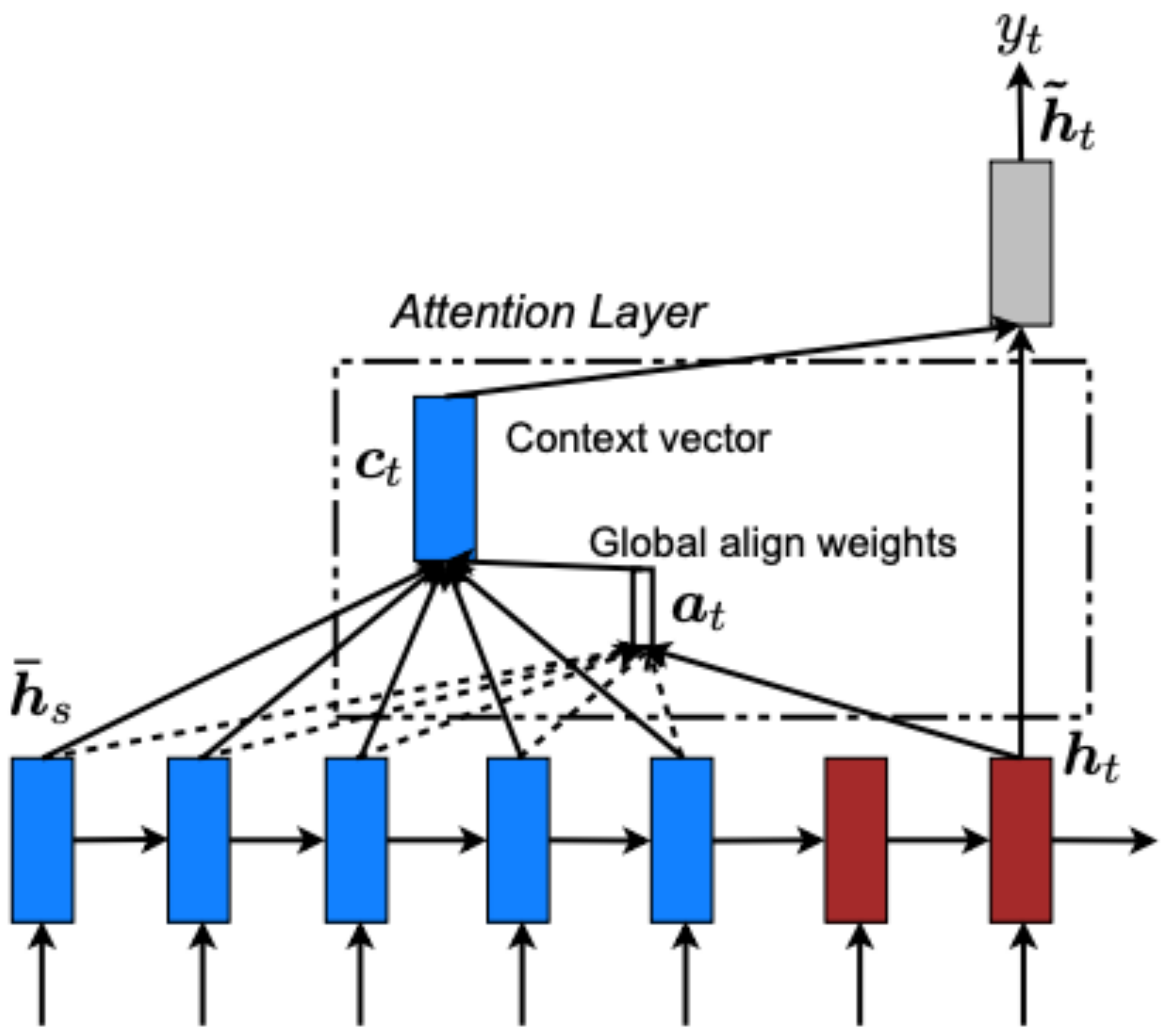
Dzmitry Bahdanau
Jacobs University Bremen, Germany

KyungHyun Cho **Yoshua Bengio***
Université de Montréal

EMNLP 2015

Effective Approaches to Attention-based Neural Machine Translation

Minh-Thang Luong **Hieu Pham** **Christopher D. Manning**
Computer Science Department, Stanford University, Stanford, CA 94305
{lmthang,hyhieu,manning}@stanford.edu

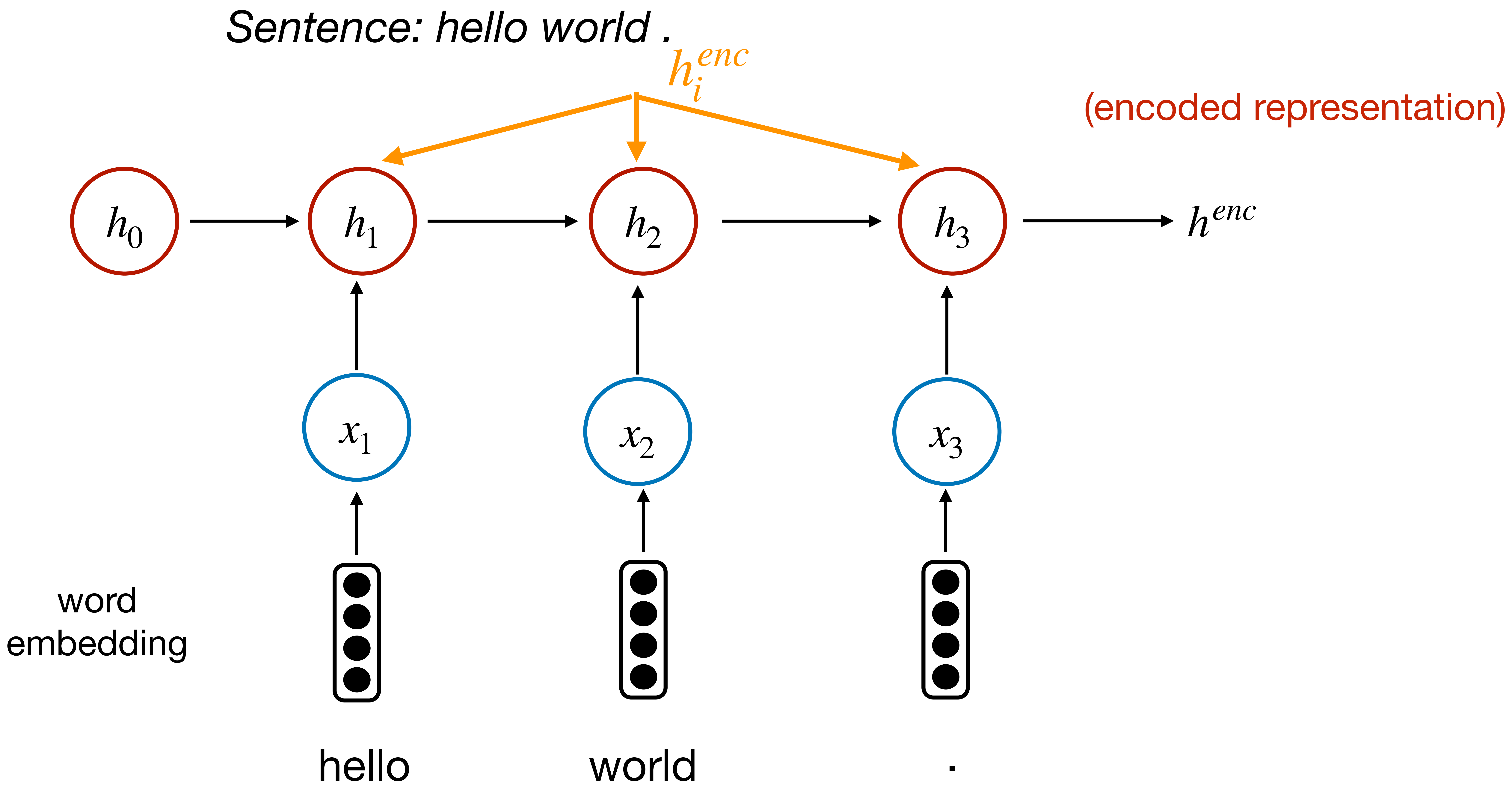


Attention

- Attention provides a solution to the bottleneck problem
- **Key idea:** At each time step during decoding, **focus on a particular part** of source sentence
  - This depends on the **decoder's** current hidden state h_t^{dec} (i.e. an idea of what you are trying to decode)
  - Usually implemented as a probability distribution over the hidden states of the **encoder** (h_i^{enc})

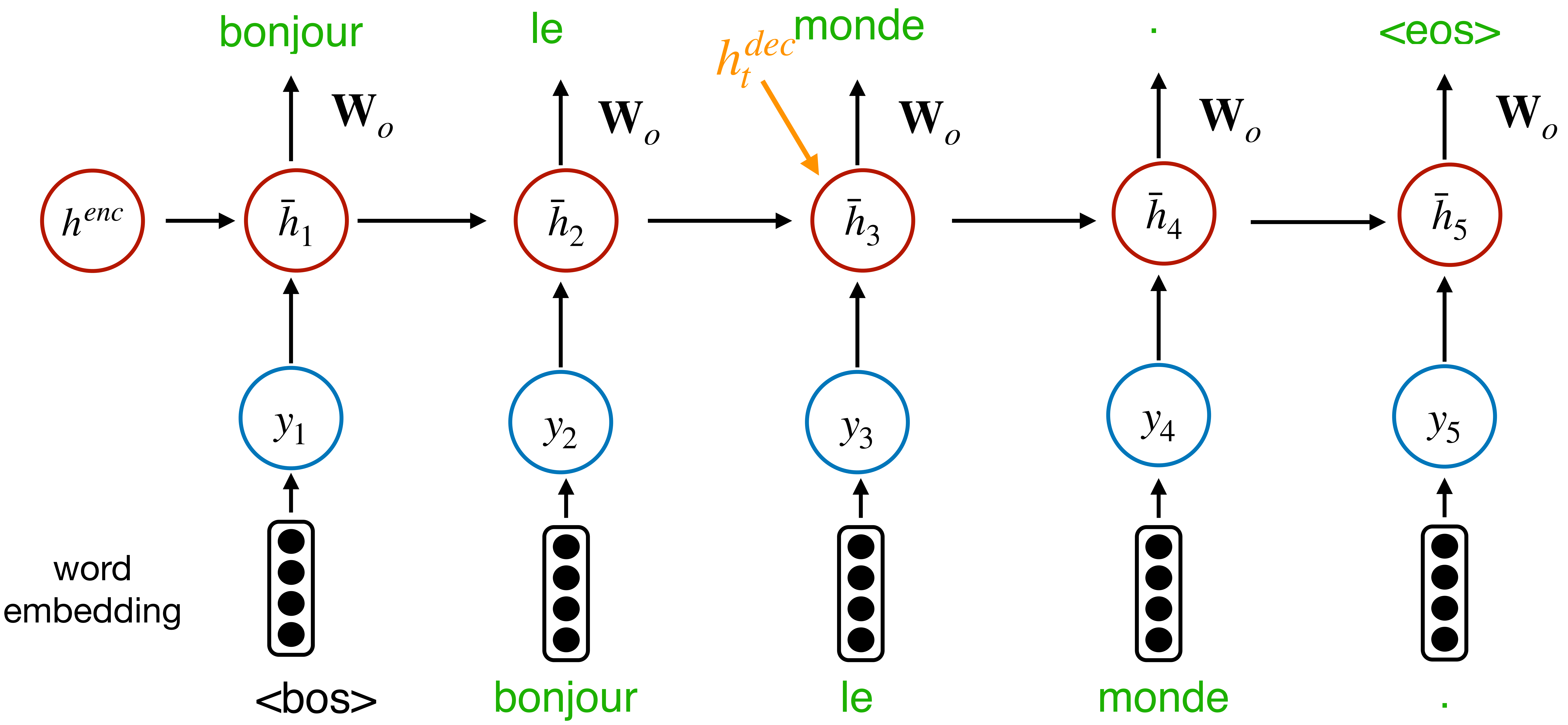
(Next lecture) Transformers = attention is all you need!

Seq2seq: Encoder

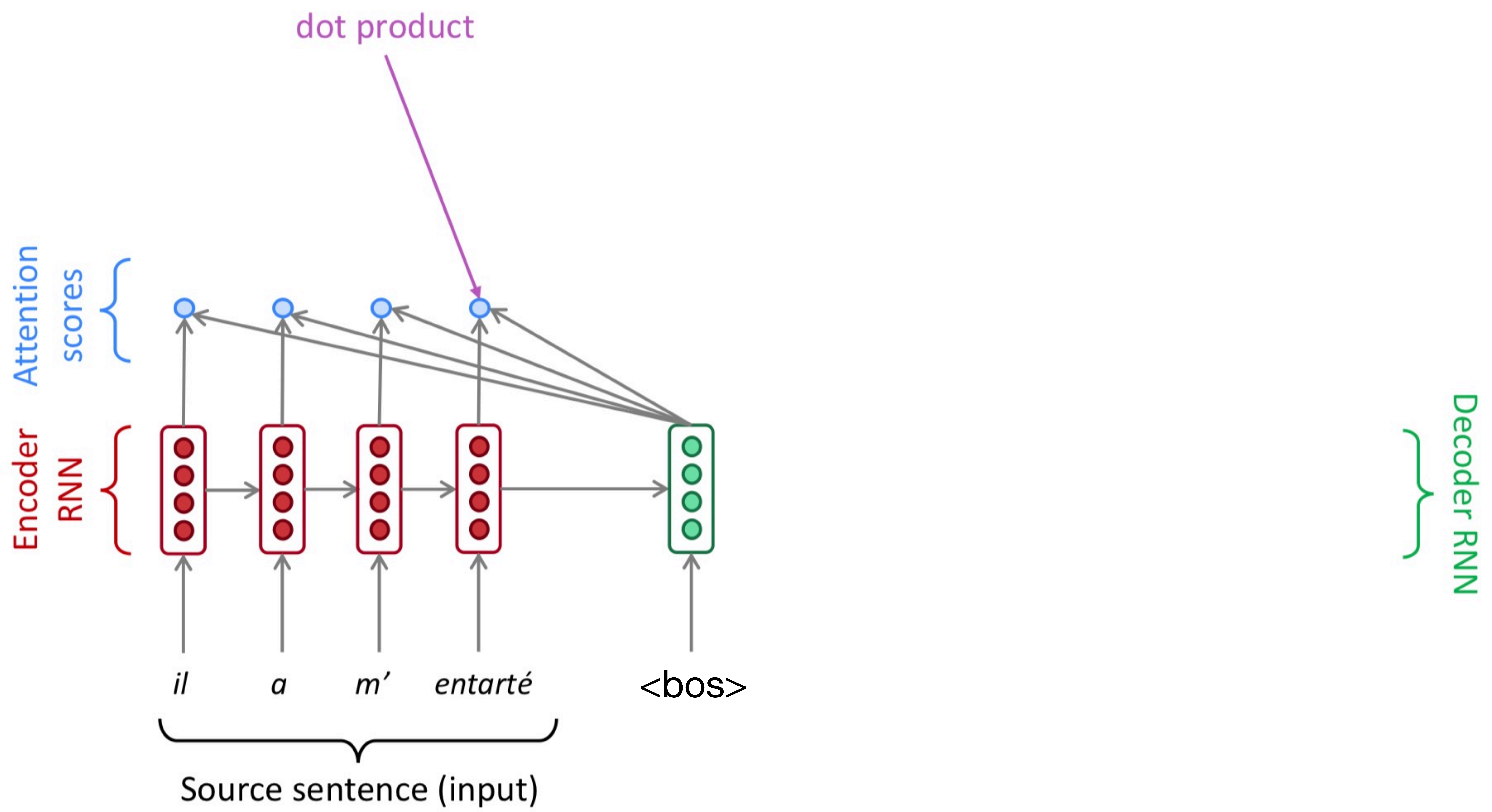


Seq2seq: Decoder

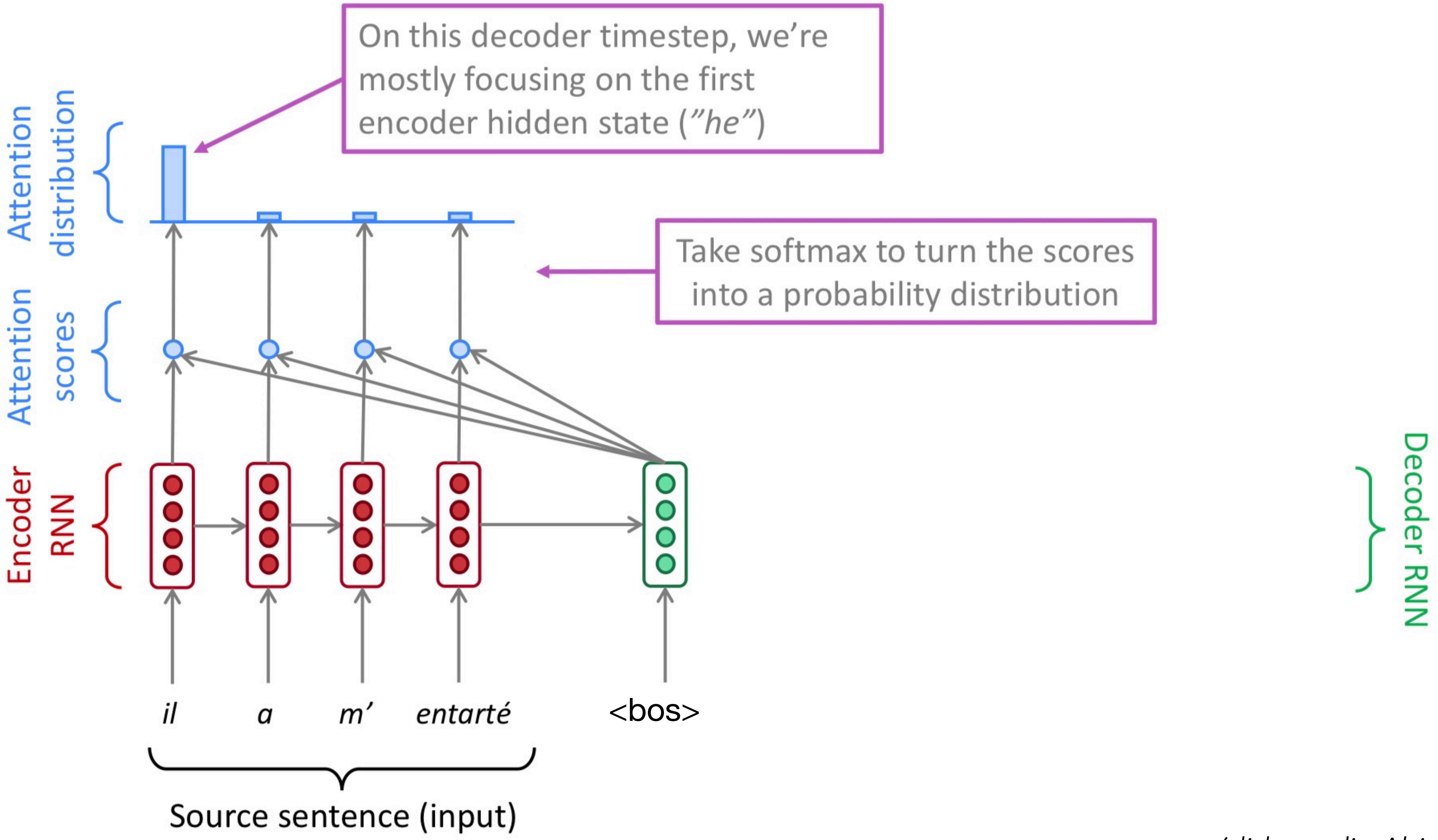
- A **conditional** language model



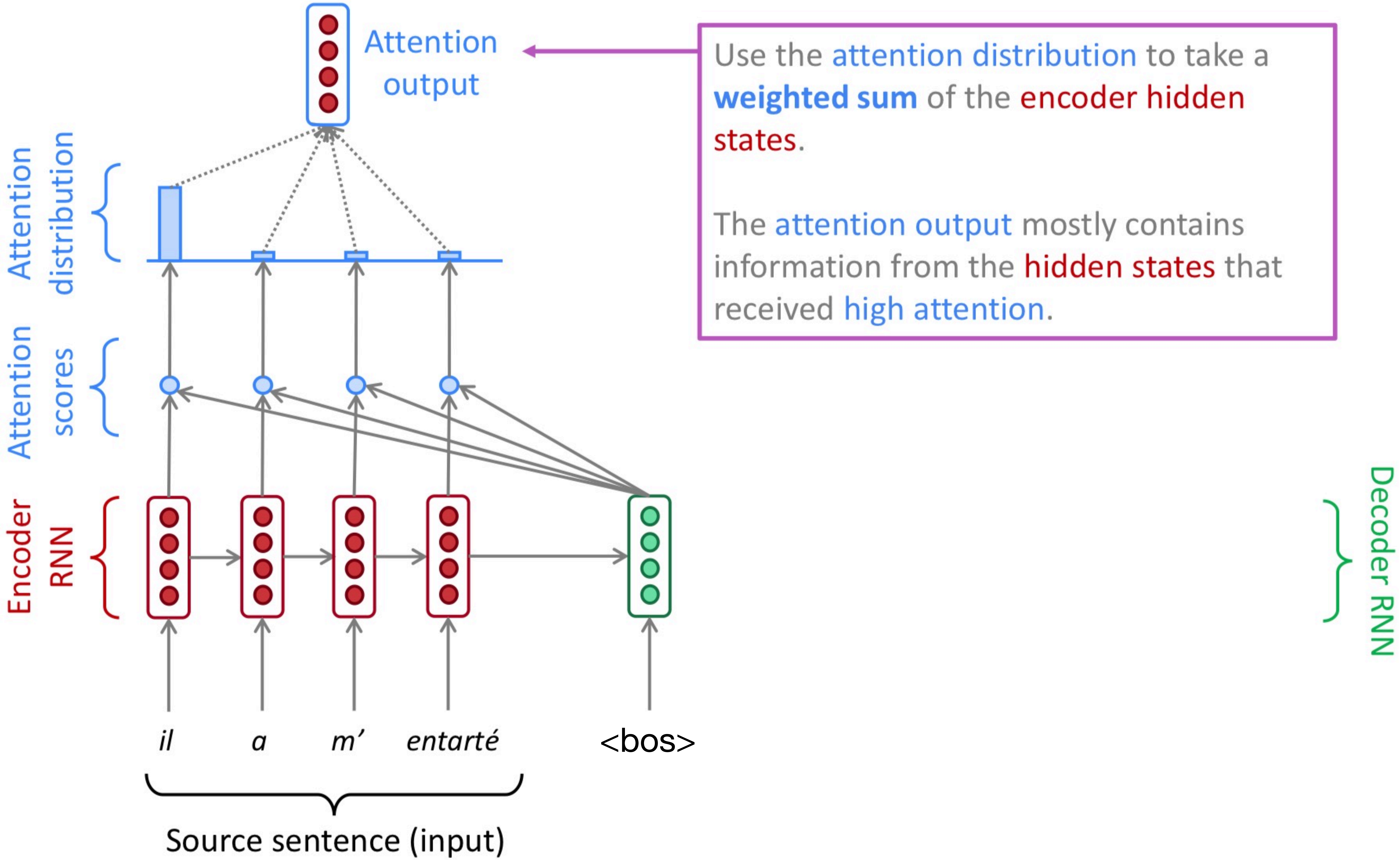
Seq2seq with attention



(slide credit: Abigail See)



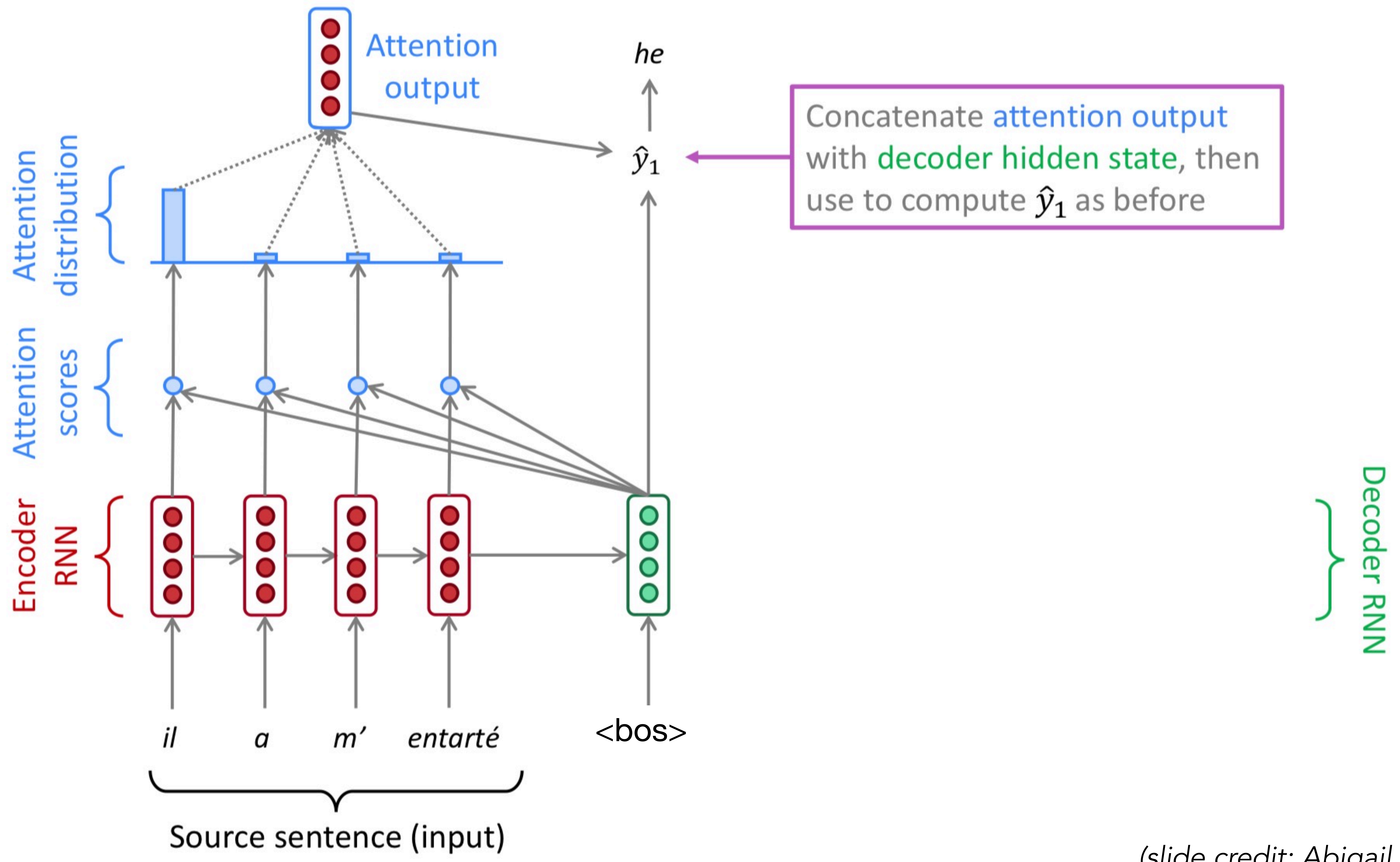
(slide credit: Abigail See)



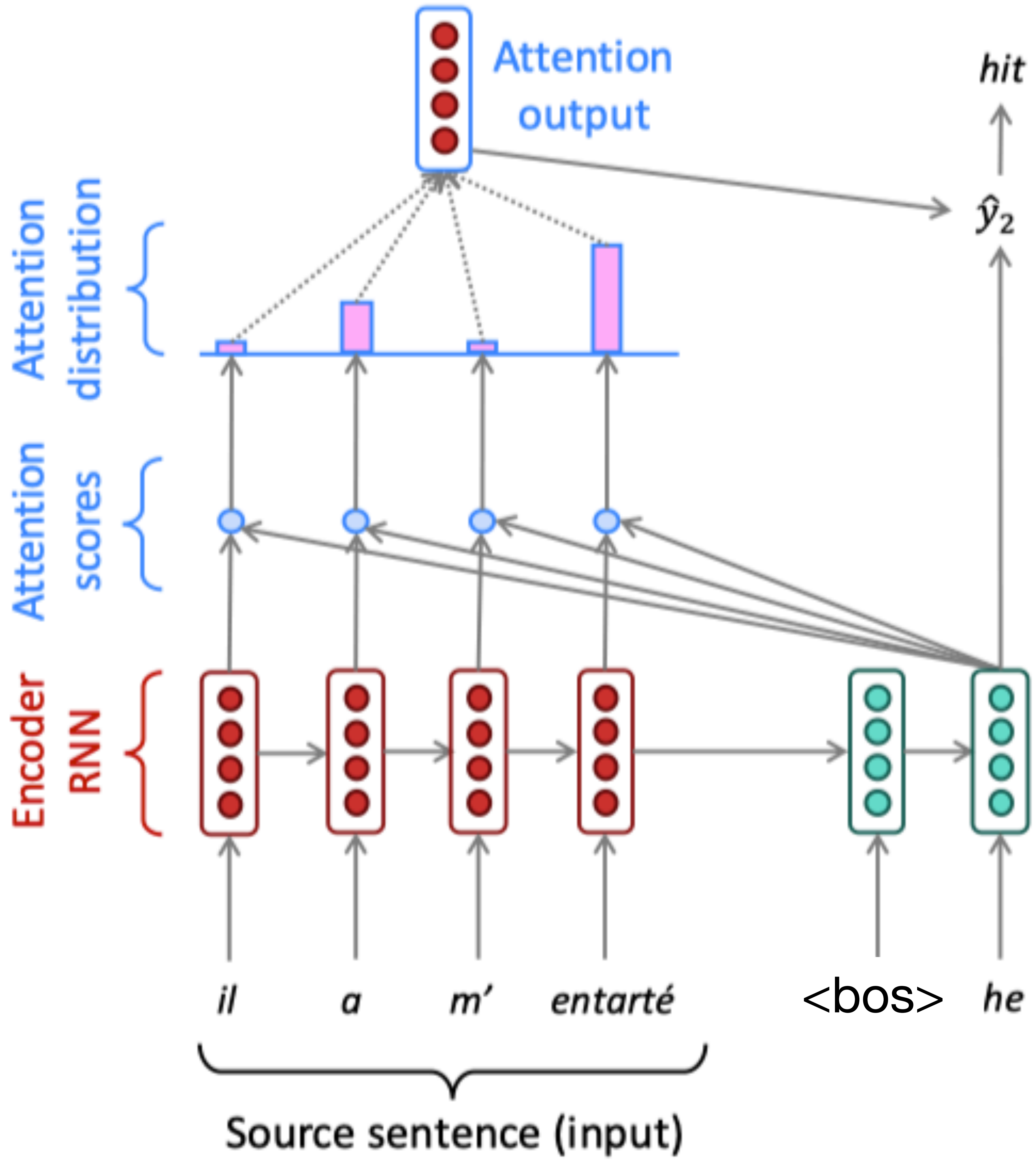
Use the attention distribution to take a **weighted sum** of the encoder hidden states.

The attention output mostly contains information from the hidden states that received high attention.

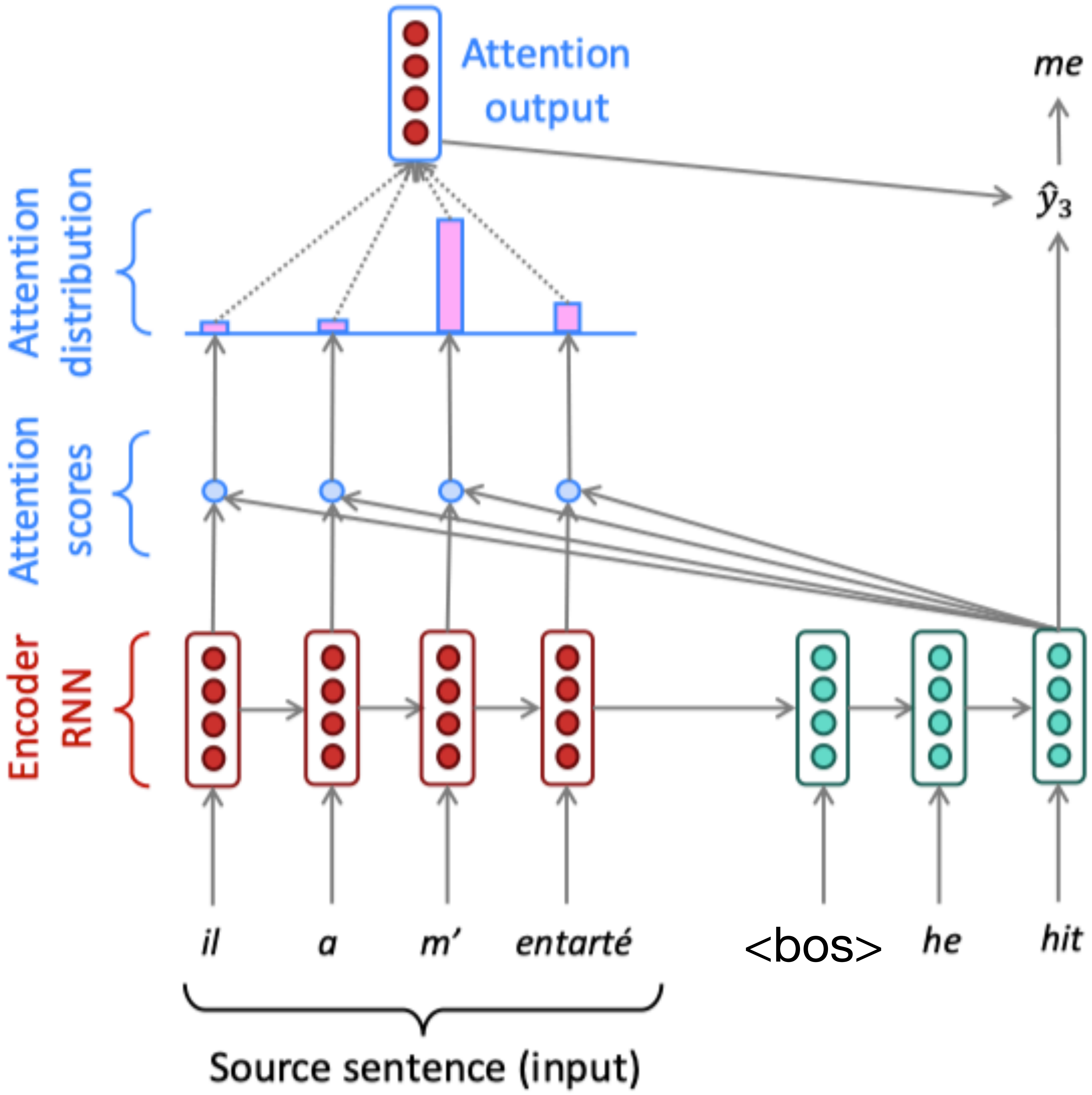
(slide credit: Abigail See)



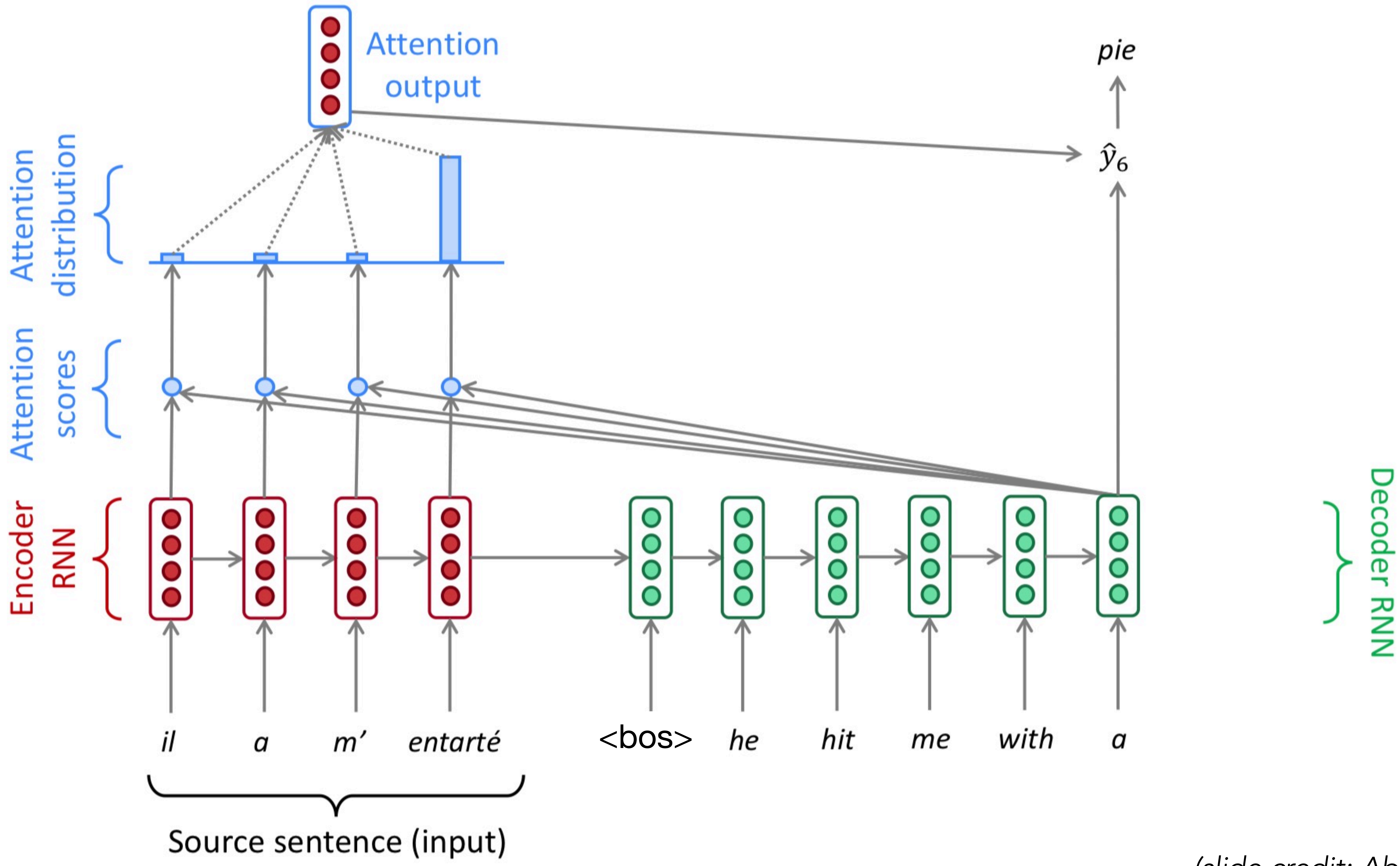
(slide credit: Abigail See)



(slide credit: Abigail See)

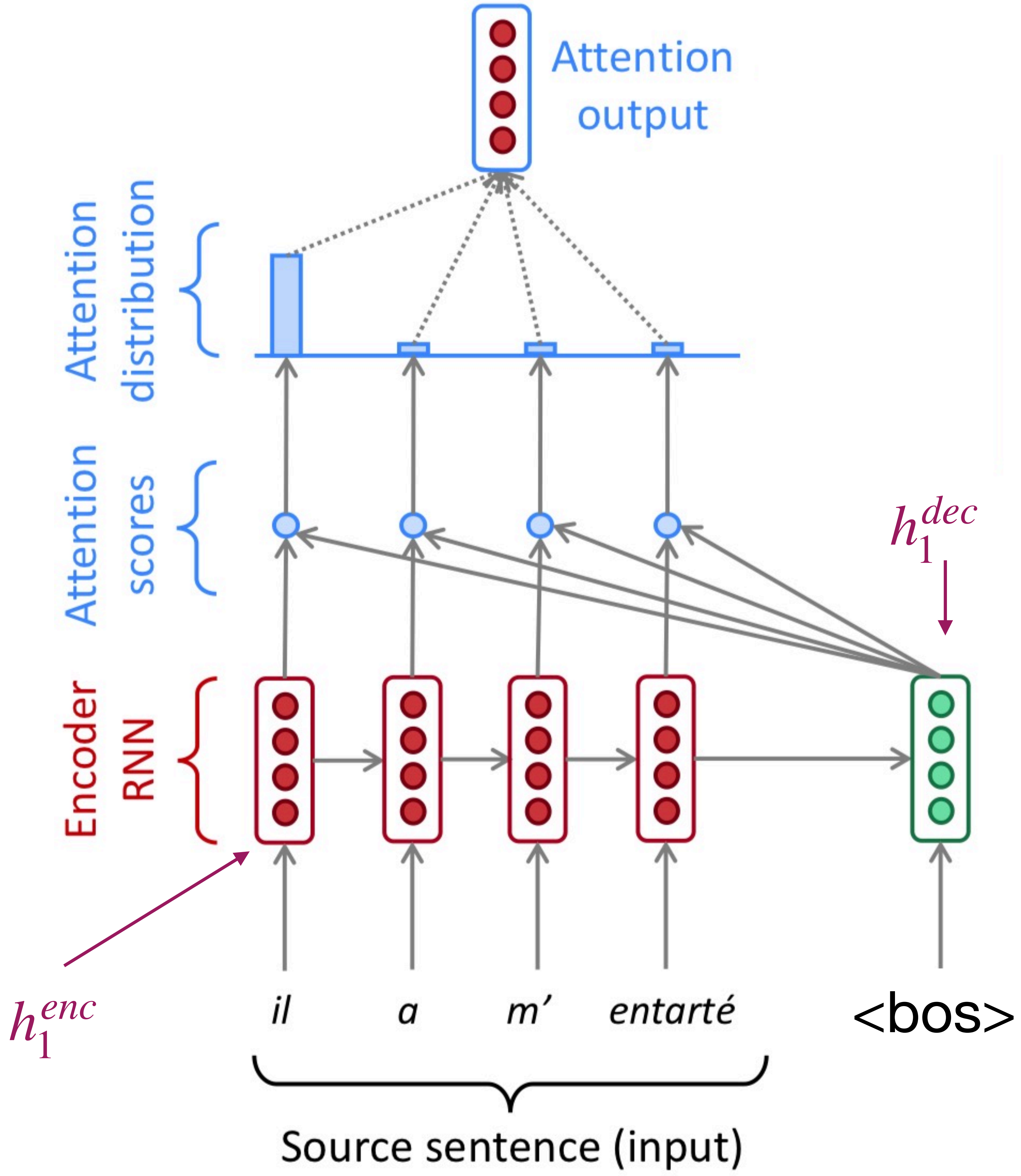


(slide credit: Abigail See)



(slide credit: Abigail See)

Computing attention



- Encoder hidden states: $h_1^{enc}, \ldots, h_n^{enc}$ (n: # of words in source sentence)
- Decoder hidden state at time t: h_t^{dec}
- First, get attention scores for this time step of decoder:

$$e^t = [g(h_1^{enc}, h_t^{dec}), \ldots, g(h_n^{enc}, h_t^{dec})]$$

- Obtain the attention distribution using softmax:

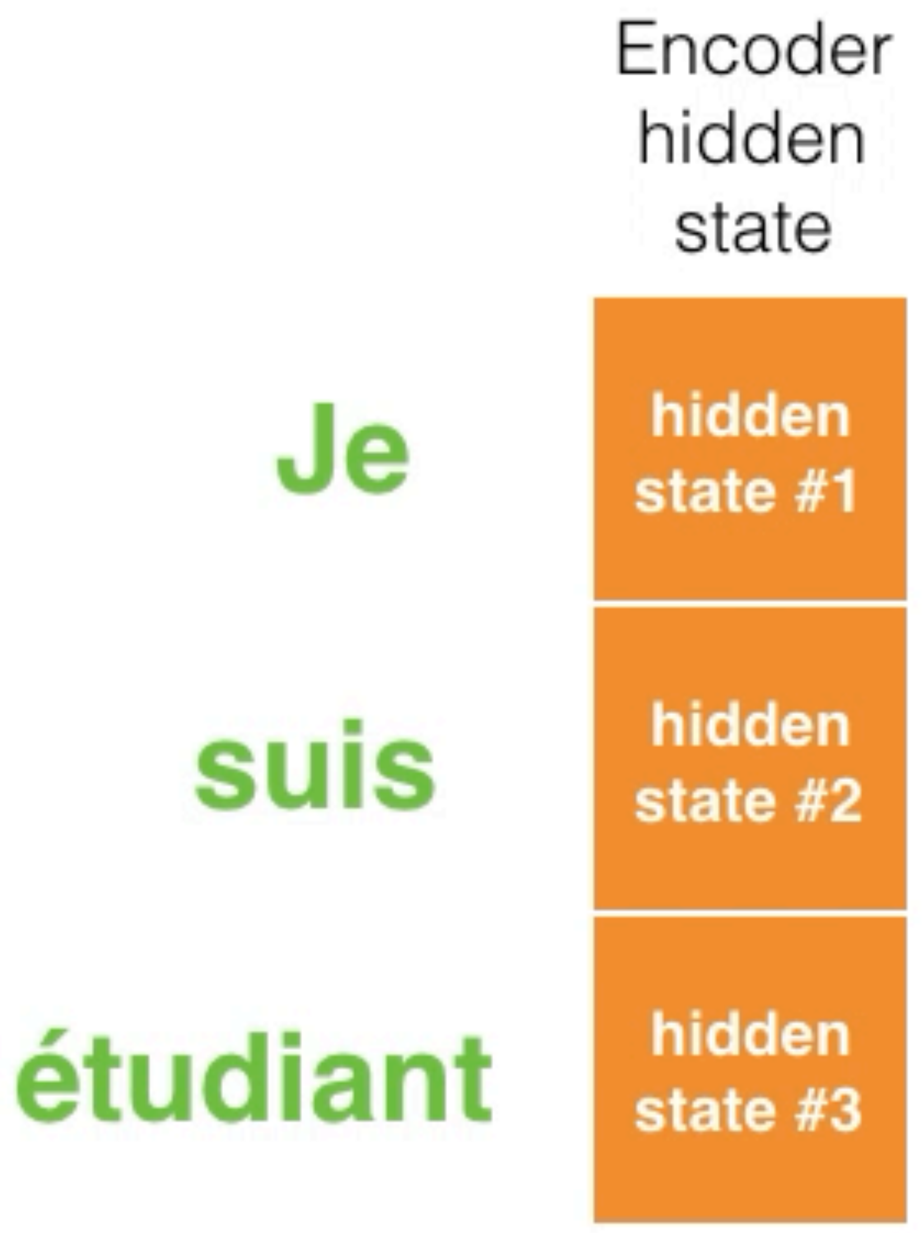
$$\alpha^t = \text{softmax}\,(e^t) \in \mathbb{R}^n$$

- Compute weighted sum of encoder hidden states:

$$a_t = \sum_{i=1}^{n} \alpha_i^t h_i^{enc} \in \mathbb{R}^h$$

- Finally, concatenate with decoder state and pass on to output layer: $\tilde{h}_t = \tanh(\mathbf{W}_c[a_t; h_t^{dec}]) \in \mathbb{R}^h \quad \mathbf{W}_c \in \mathbb{R}^{2h \times h}$

$$\hat{\mathbf{y}}_t = \text{softmax}(\mathbf{W}_o \tilde{h}_t)$$



https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

(credits: Jay Alammar)

Types of attention

- Assume encoder hidden states $h_1^{enc}, h_2^{enc}, \ldots, h_n^{enc}$ and a decoder hidden state h_t^{dec}

1. **Dot-product attention** (assumes equal dimensions for h^{enc} and h_t^{dec}):

$$g(h_i^{enc}, h_t^{dec}) = (h_t^{dec})^T \; h_i^{enc} \in \mathbb{R}$$

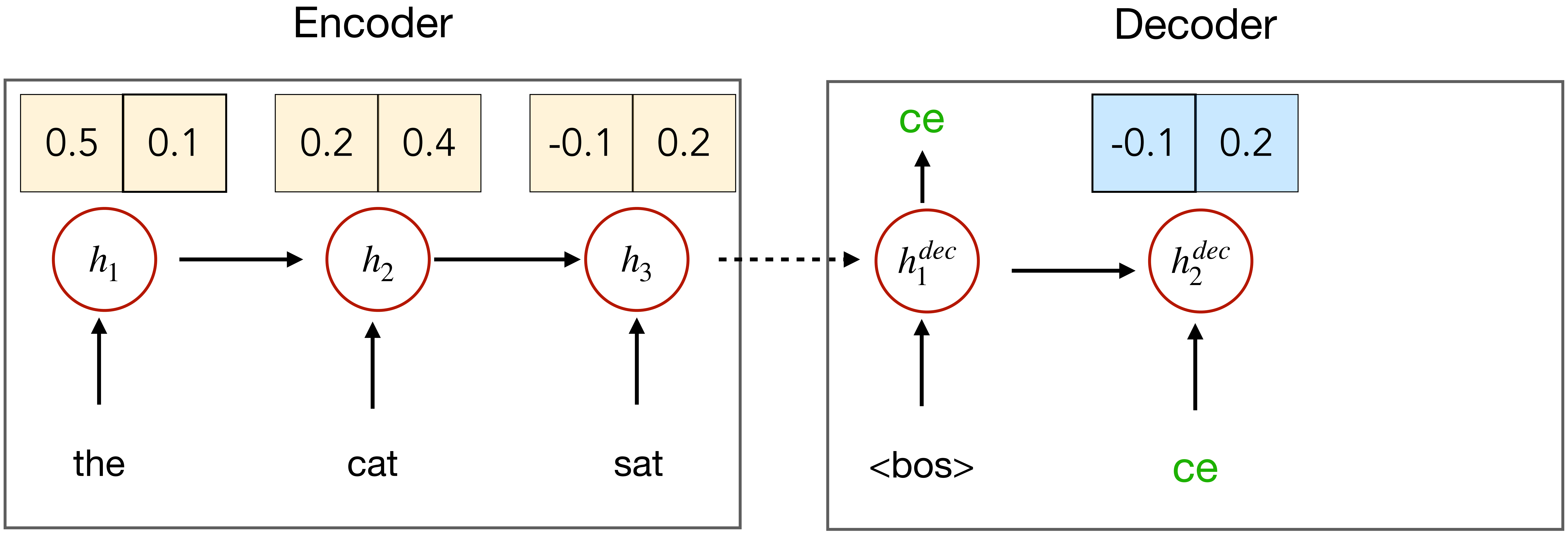
2. **Multiplicative attention:**

$$g(h_i^{enc}, h_t^{dec}) = (h_t^{dec})^T \; W \; h_i^{enc} \in \mathbb{R}$$, where W is a weight matrix (learned)

3. **Additive attention:**

$$g(h_i^{enc}, h_t^{dec}) = v^T \tanh\left(W_1 h_i^{enc} + W_2 h_t^{dec}\right) \in \mathbb{R}$$

where W_1, W_2 are weight matrices (learned) and v is a weight vector (learned)



Dot-product attention:

$$g(h_i^{enc}, h_t^{dec}) = h_t^{dec} \cdot h_i^{enc}$$

Assuming we use dot product attention, which input word will have the highest attention value at current time step?

A) the
B) cat
C) sat

The answer is (B)

the: -0.05 + 0.02
cat: -0.02 + 0.08
sat: 0.01 + 0.04

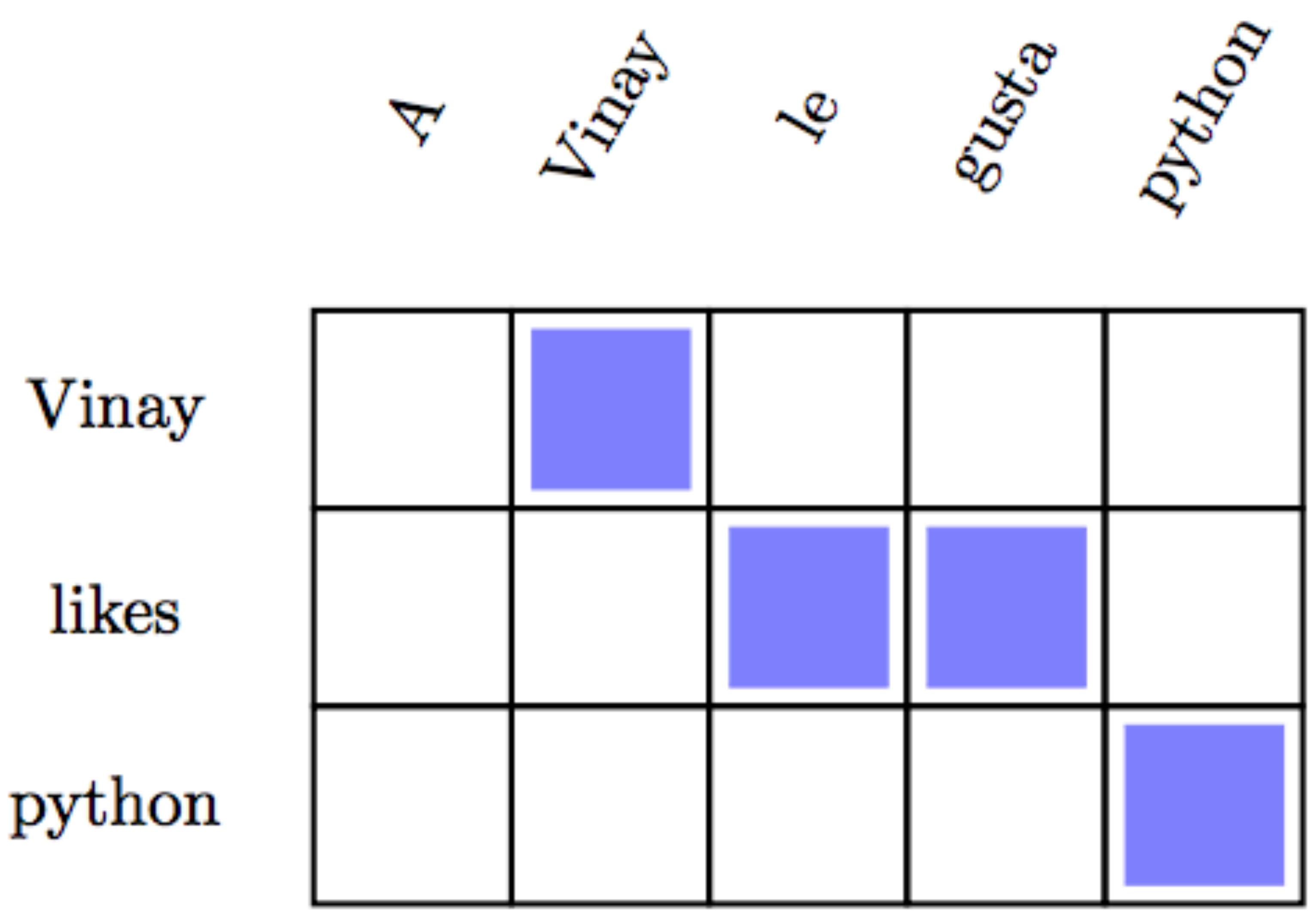
Attention improves translation

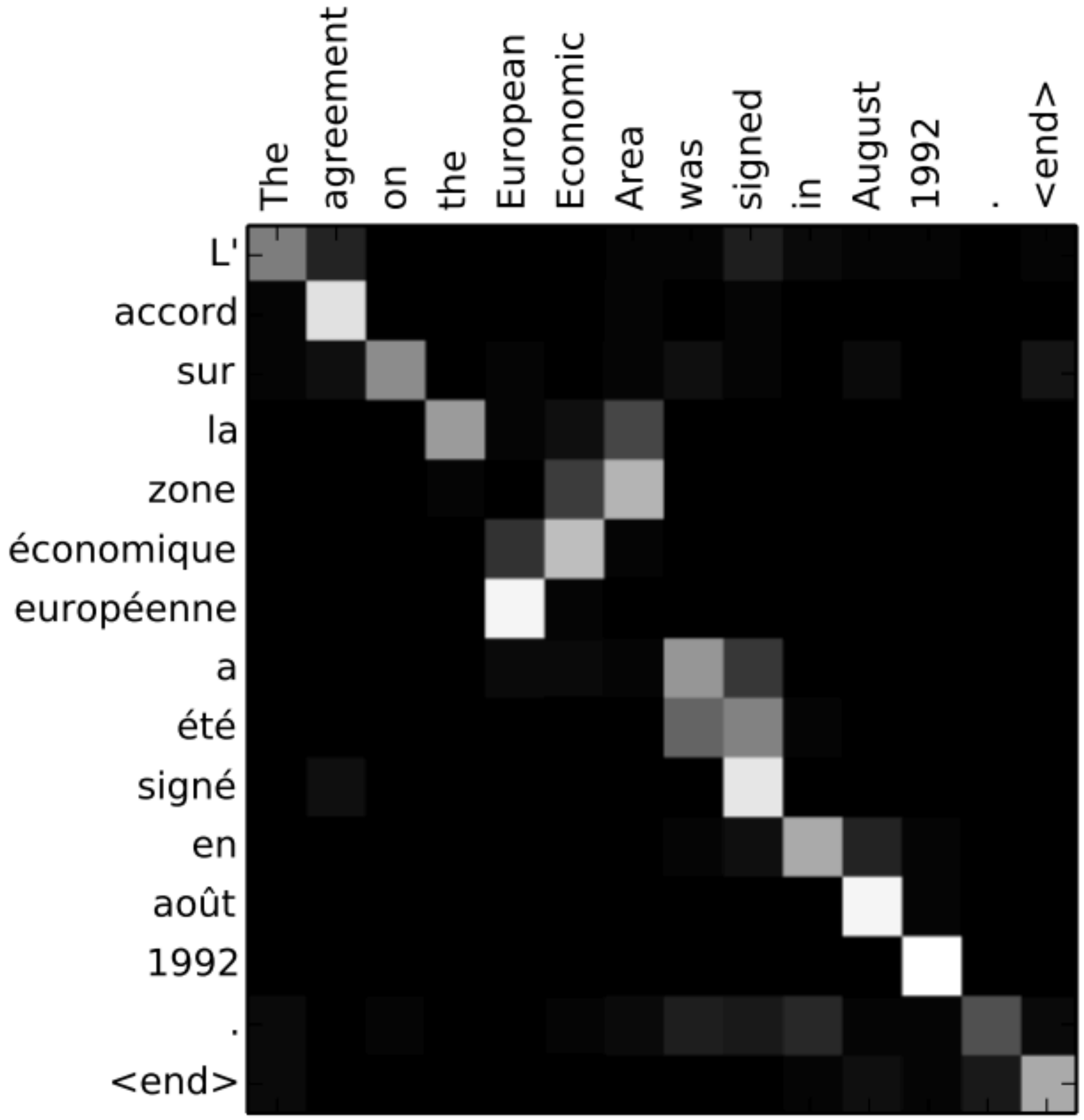
System	Ppl	BLEU
Winning WMT'14 system – *phrase-based + large LM* (Buck et al., 2014)		20.7
Existing NMT systems		
RNNsearch (Jean et al., 2015)		16.5
RNNsearch + unk replace (Jean et al., 2015)		19.0
RNNsearch + unk replace + large vocab + *ensemble* 8 models (Jean et al., 2015)		**21.6**
Our NMT systems		
Base	10.6	11.3
Base + reverse	9.9	12.6 (+*1.3*)
Base + reverse + dropout	8.1	14.0 (+*1.4*)
Base + reverse + dropout + global attention (*location*)	7.3	16.8 (+*2.8*)
Base + reverse + dropout + global attention (*location*) + feed input	6.4	18.1 (+*1.3*)
Base + reverse + dropout + local-p attention (*general*) + feed input	5.9	19.0 (+*0.9*)
Base + reverse + dropout + local-p attention (*general*) + feed input + unk replace		20.9 (+*1.9*)
Ensemble 8 models + unk replace		**23.0** (+*2.1*)

(Luong et al., 2015)

Visualizing attention

Recall: alignment





(credits: Jay Alammar)