UNIVERSIDADE CATÓLICA PORTUGUESA

BRAGA

Deep Learning

Session 18

Introduction to Attention

Applied Data Science

2024/2025

Recap: RNNs

output distribution

$$\hat{y} = \mathrm{softmax}(W_2 h^{(t)} + b_2)$$

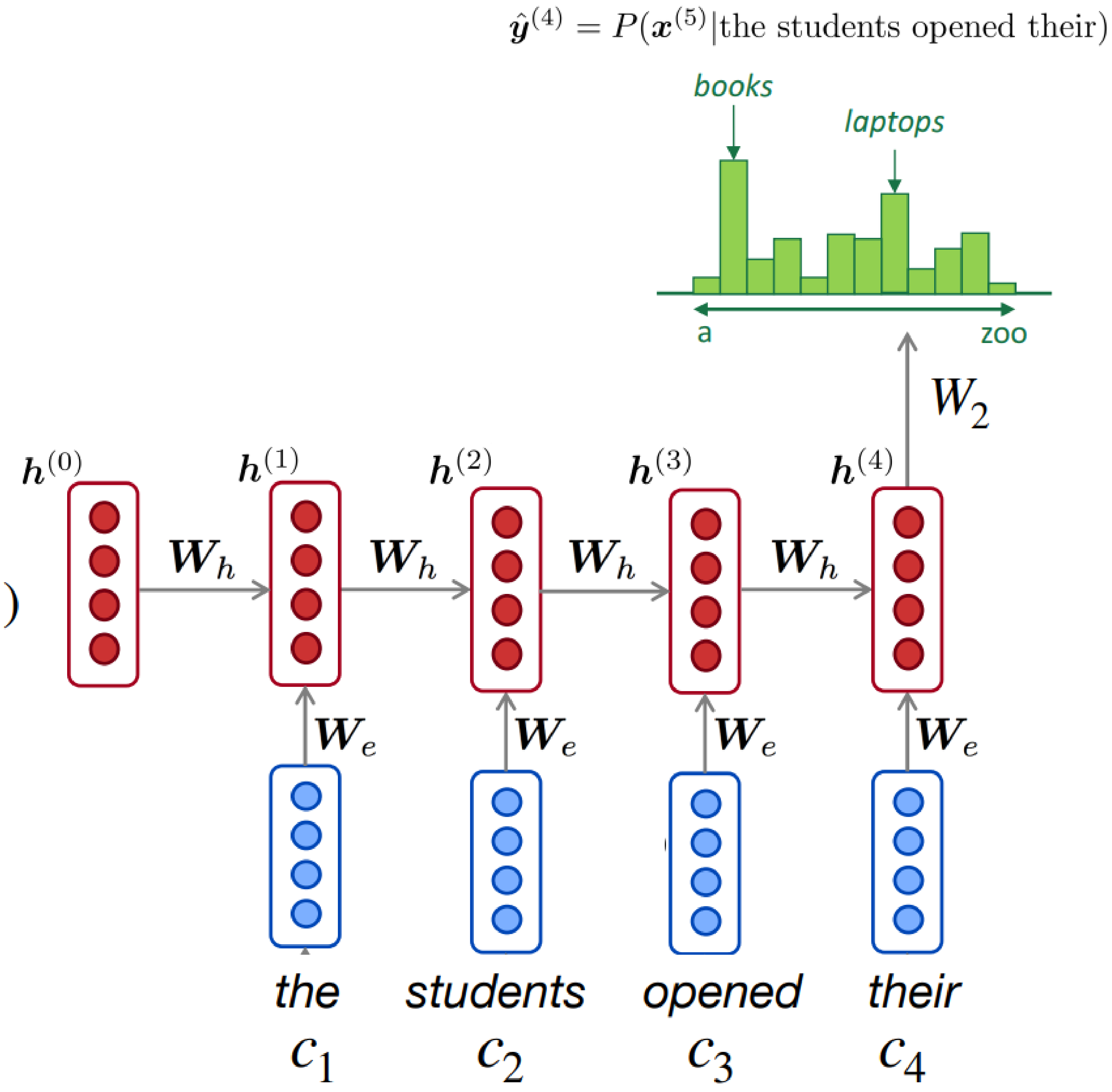
hidden states

$$h^{(t)} = f(W_h h^{(t-1)} + W_e c_t + b_1)$$

$h^{(0)}$ is initial hidden state!

word embeddings

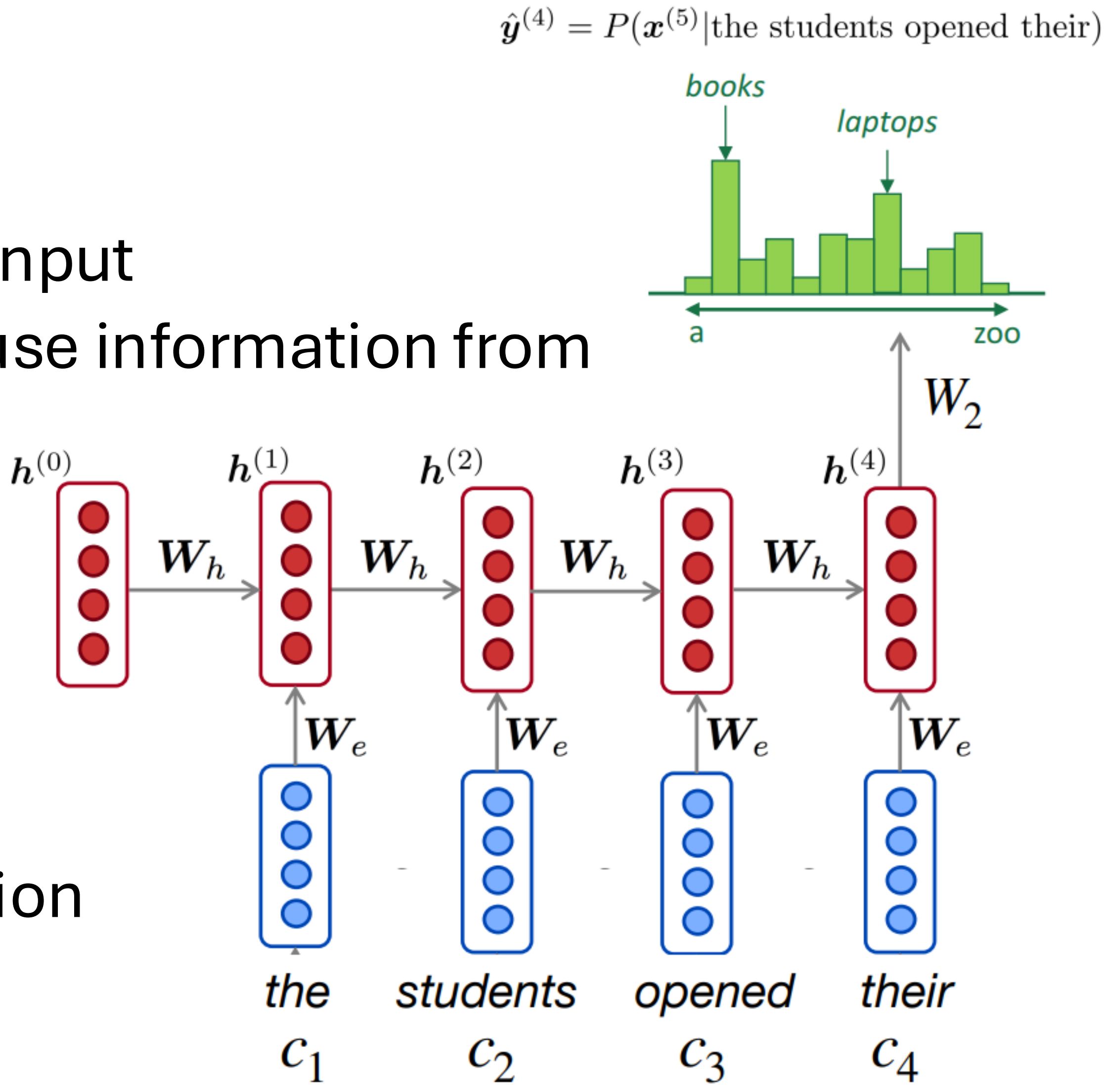
$$c_1, c_2, c_3, c_4$$



Recap: RNNs

- RNN Advantages:
  - Can process any length input
  - Model size doesn't increase for longer input
  - Computation for step t can (in theory) use information from many steps back.
  - Weights are shared across timesteps

- RNN Disadvantages:
  - Recurrent computation is slow
  - In practice, difficult to access information from many steps back.



Recap: Training a RNN Language Model

- Get a **big corpus** of text which is a sequence of words $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$

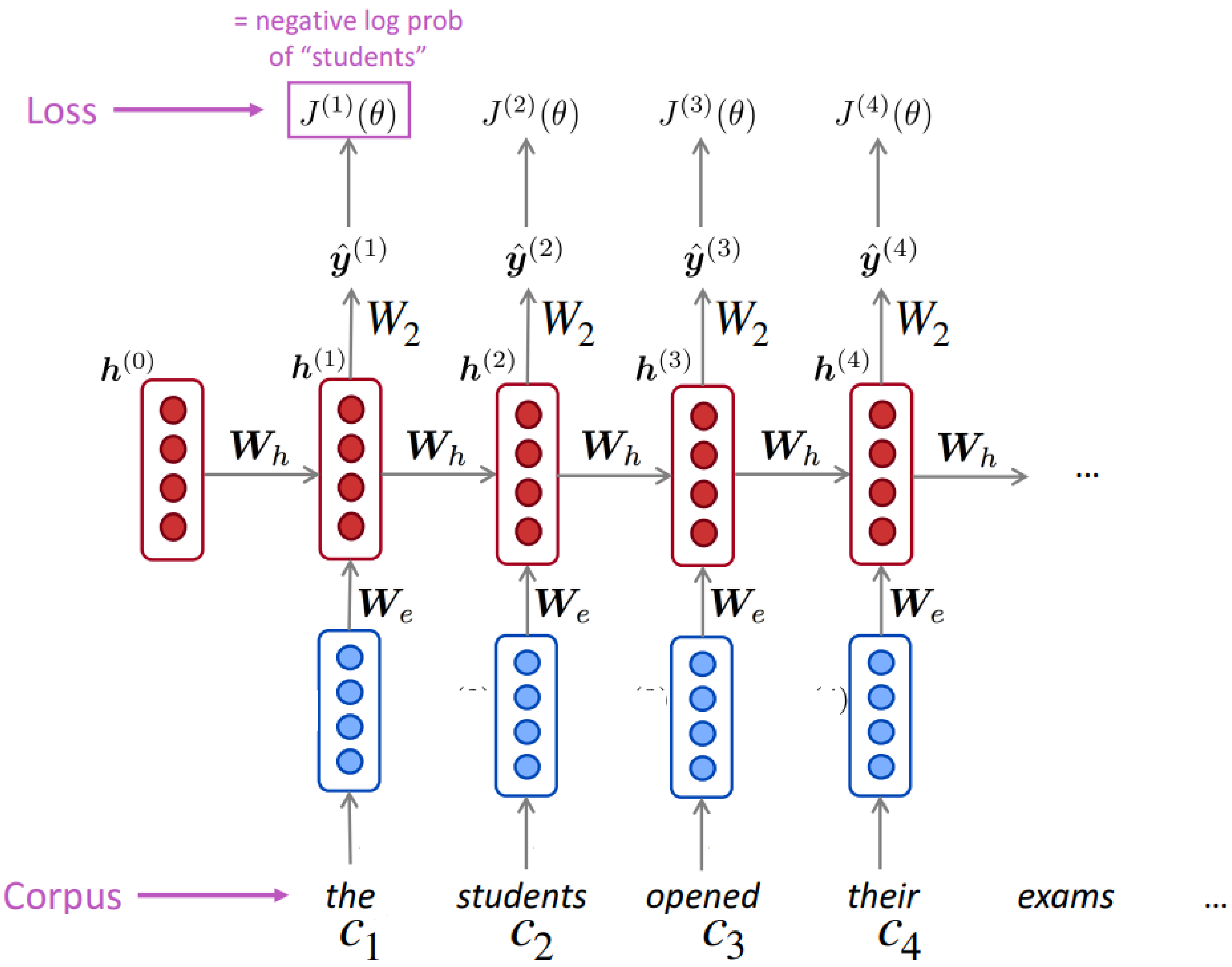
- Fed it into the RNN. Compute the outpur distribution $\hat{\boldsymbol{y}}^{(t)}$ for **every step t**.
  - i.e. predict the probability distribution of every word given the words so far

- **Loss funtion** on step t is usualy cross-entropy between the predicted probability distribution $\hat{\boldsymbol{y}}^{(t)}$, and the true next word $\boldsymbol{y}^{(t)} = \boldsymbol{x}^{(t+1)}$:

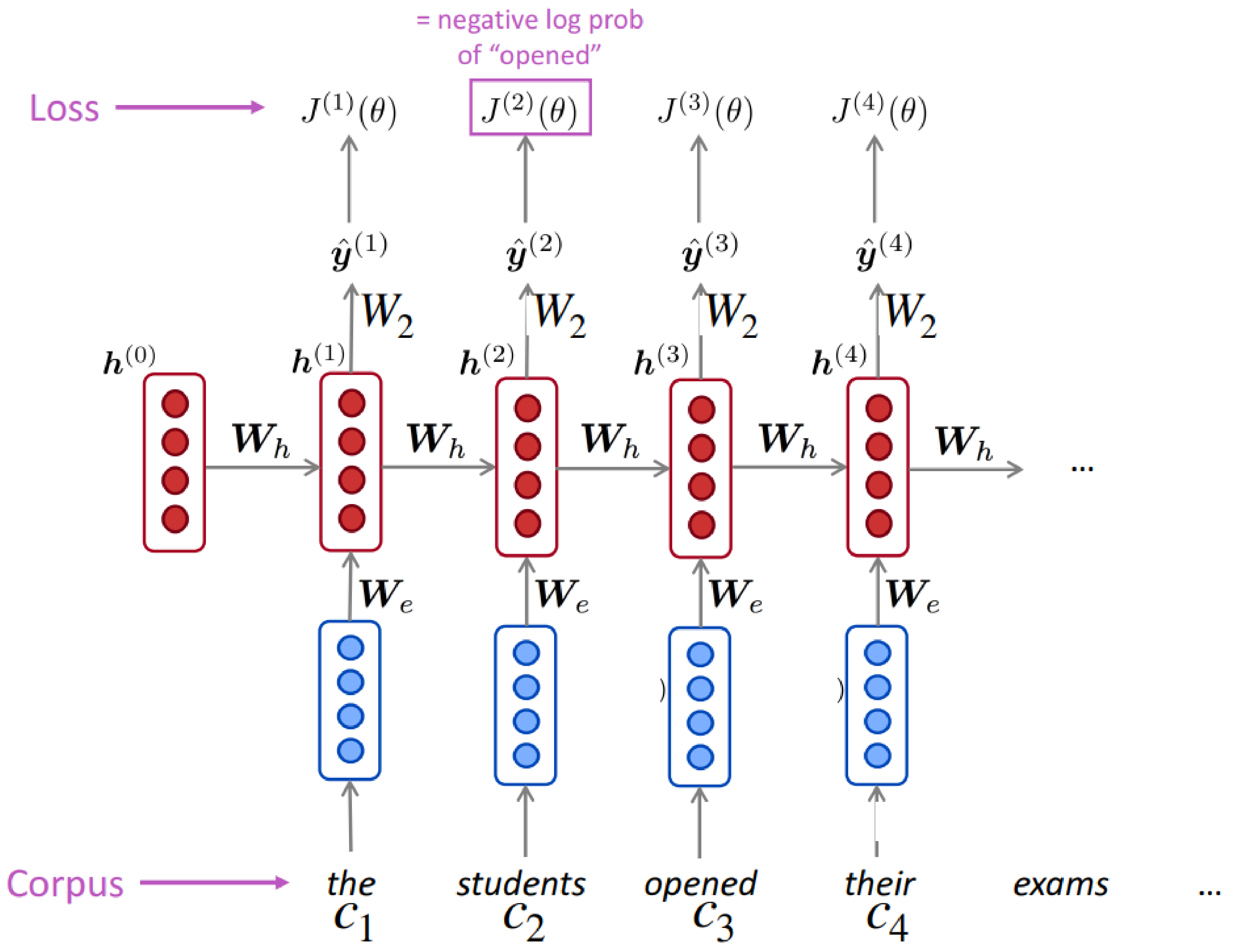
$$J^{(t)}(\theta) = CE(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}) = -\sum_{j=1}^{|V|} y_j^{(t)} \log \hat{y}_j^{(t)} \Longrightarrow J(\theta) = \frac{1}{T}\sum_{t=1}^{T} J^{(t)}(\theta)$$

Average to get the **overall loss**!

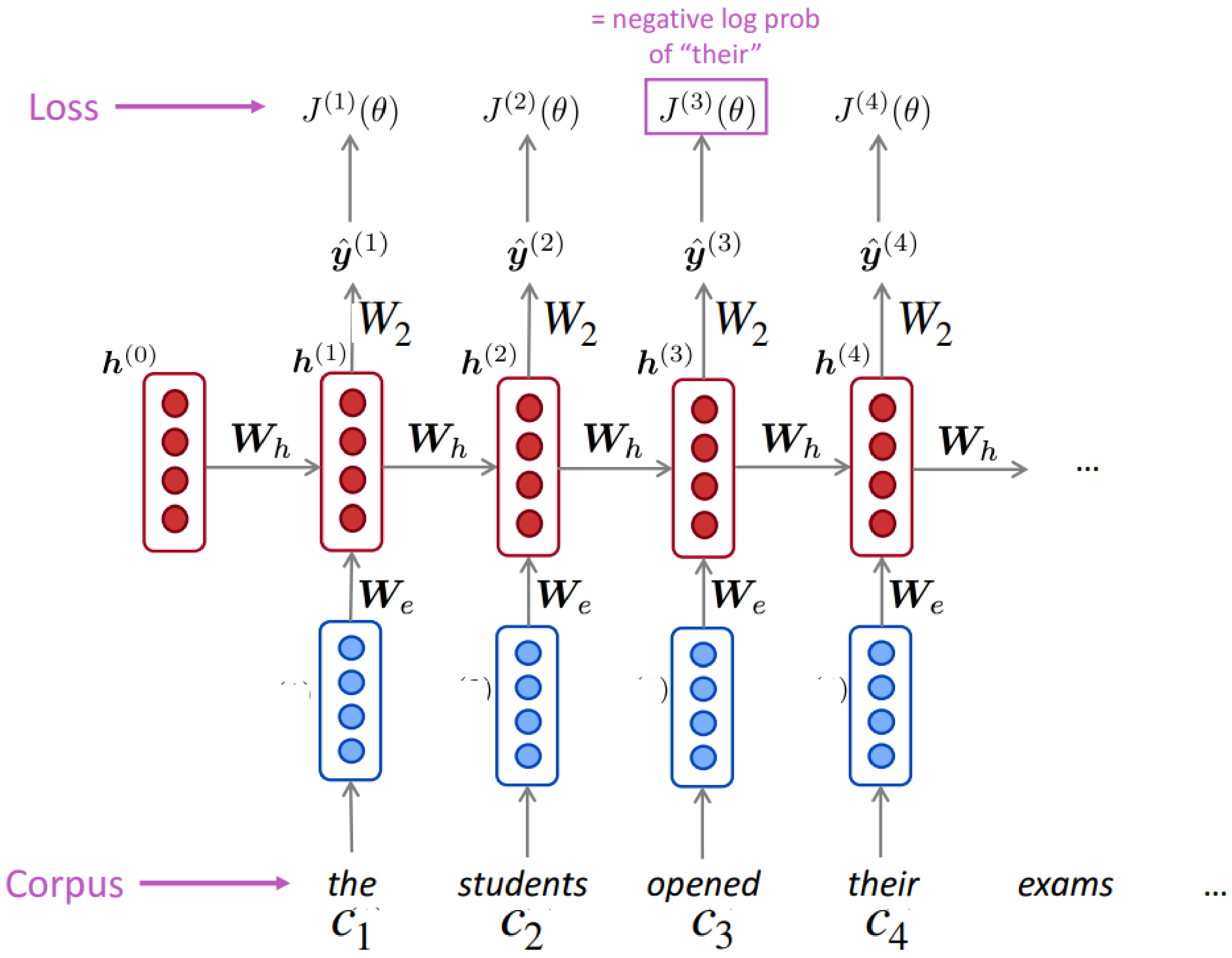
Recap: Training a RNN Language Model



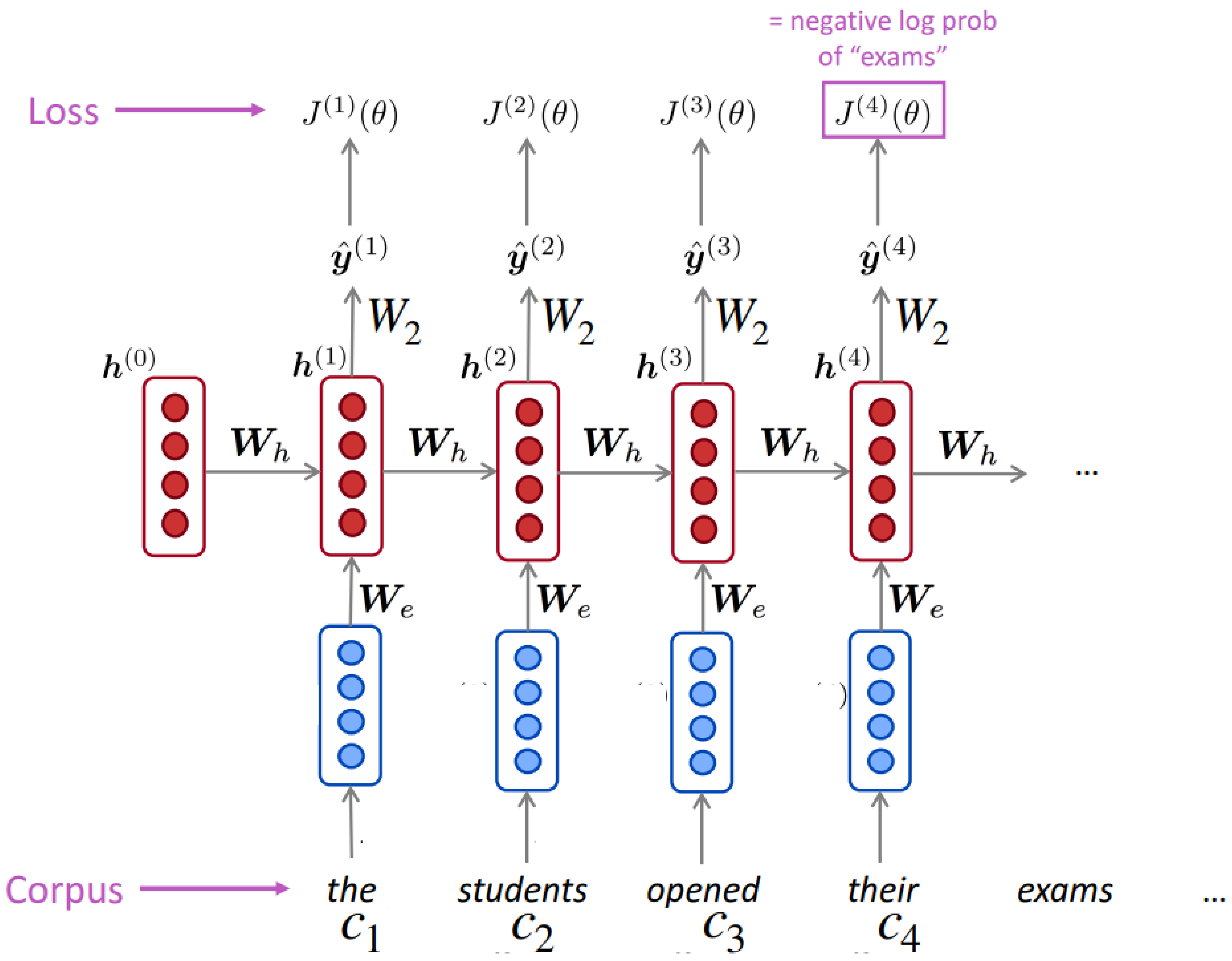
Recap: Training a RNN Language Model



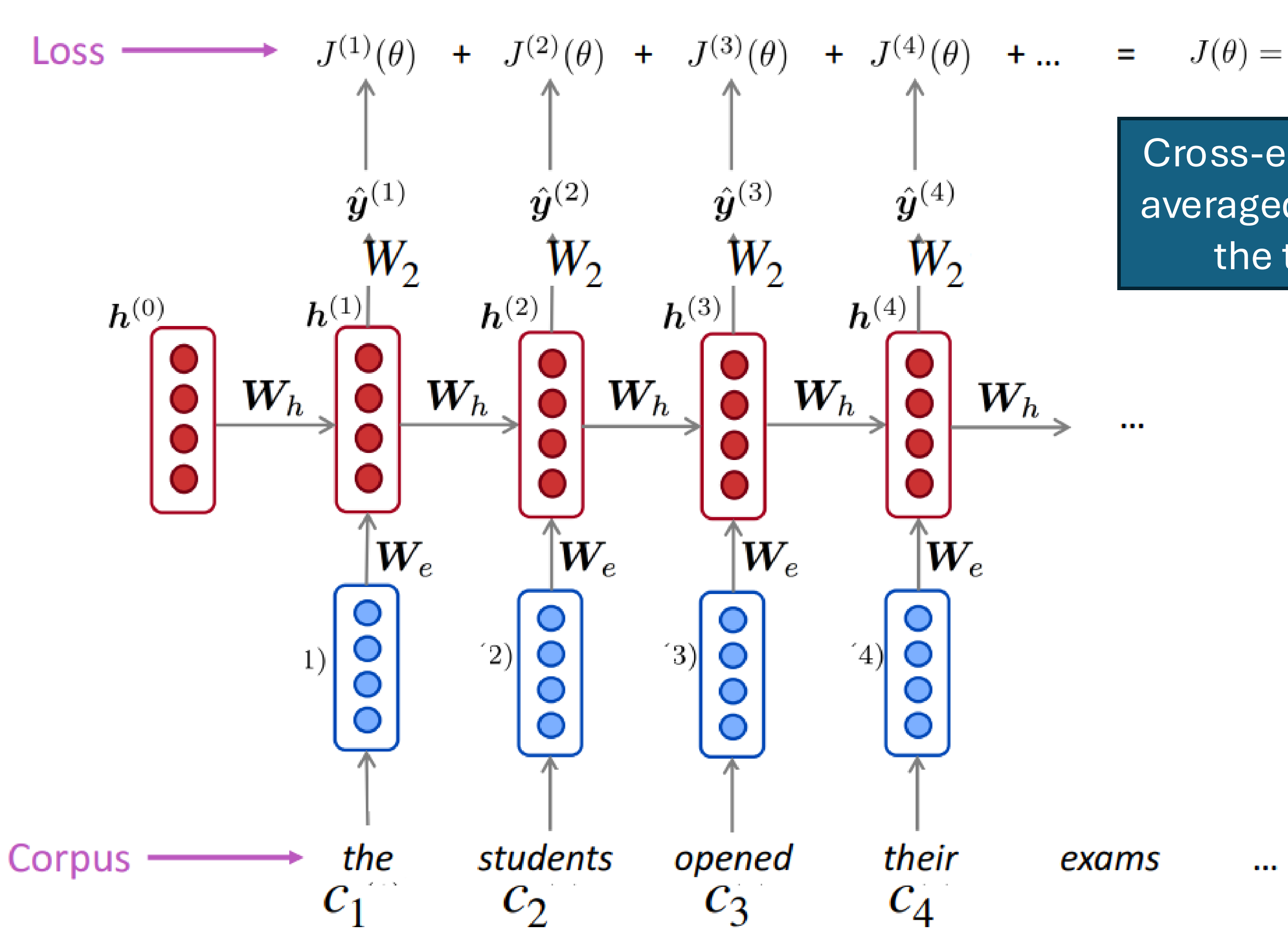
Recap: Training a RNN Language Model



Recap: Training a RNN Language Model



Recap: Training a RNN Language Model

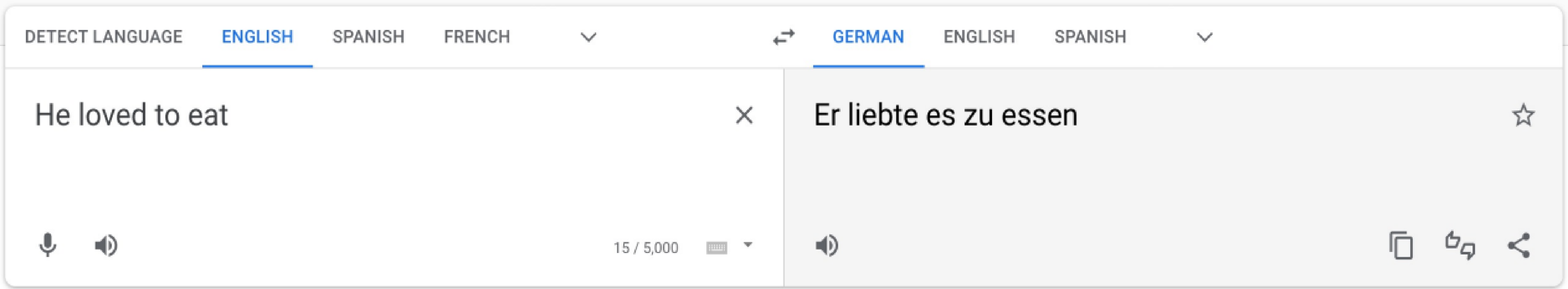


$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T} J^{(t)}(\theta)$$

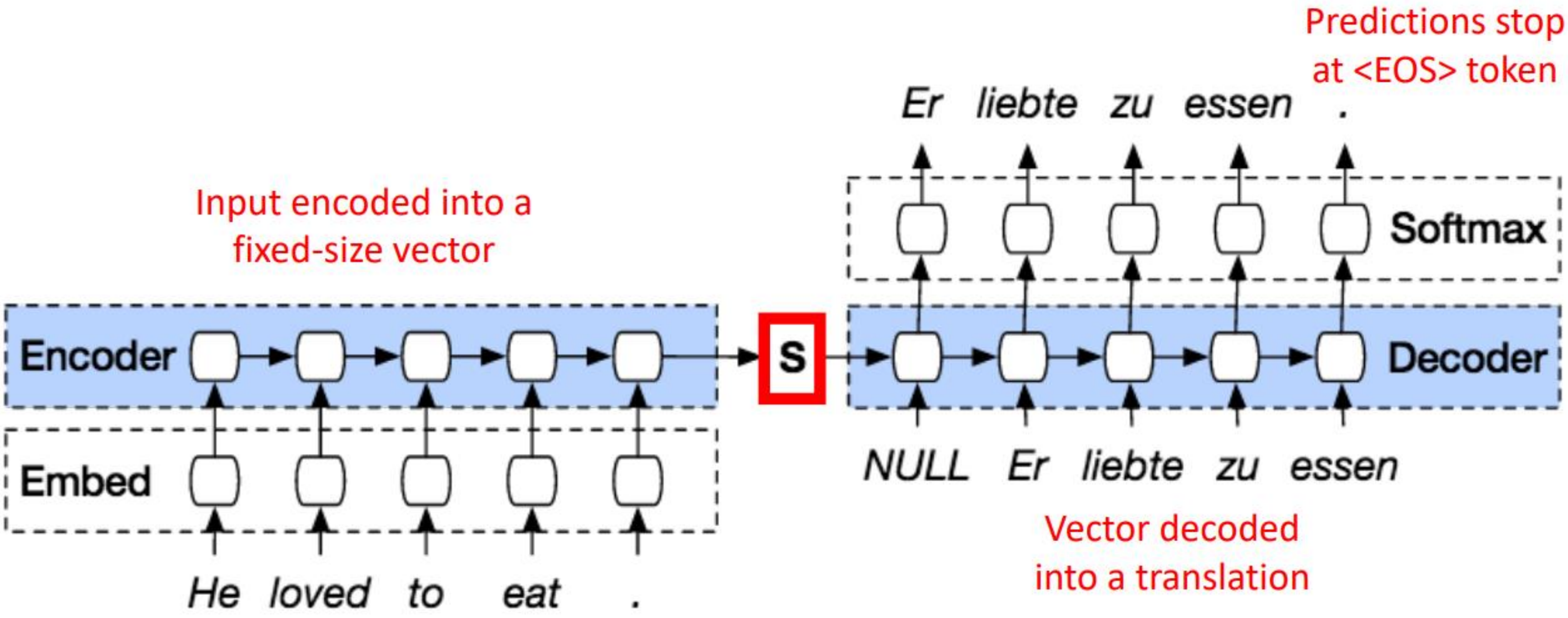
Cross-entropy loss averaged over all of the tokens.

Task: Machine Translation

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA



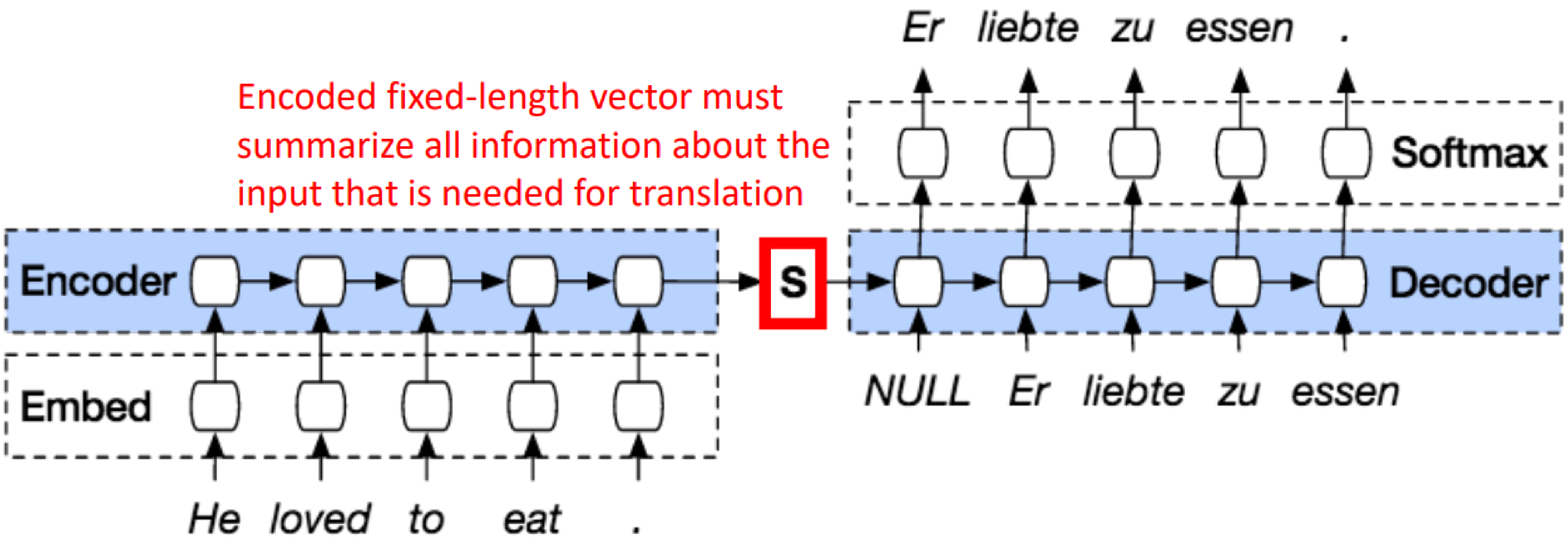
Task: Machine Translation



Task: Machine Translation

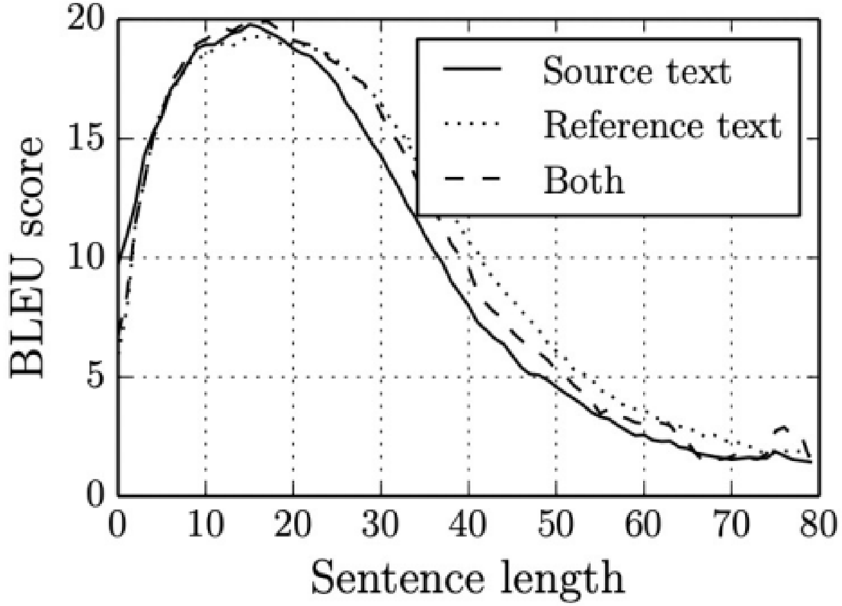
UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

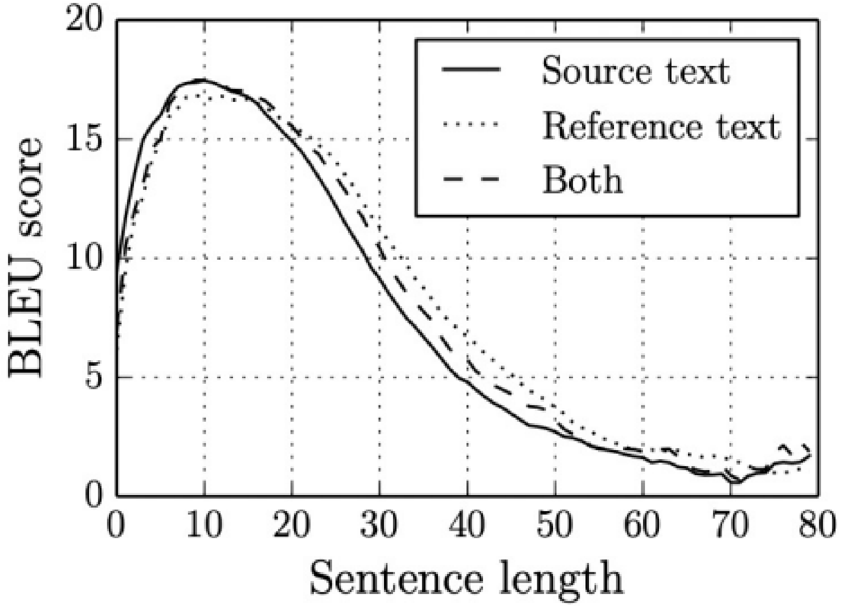
Encoded fixed-length vector must summarize all information about the input that is needed for translation



Task: Machine Translation

(larger scores are better)

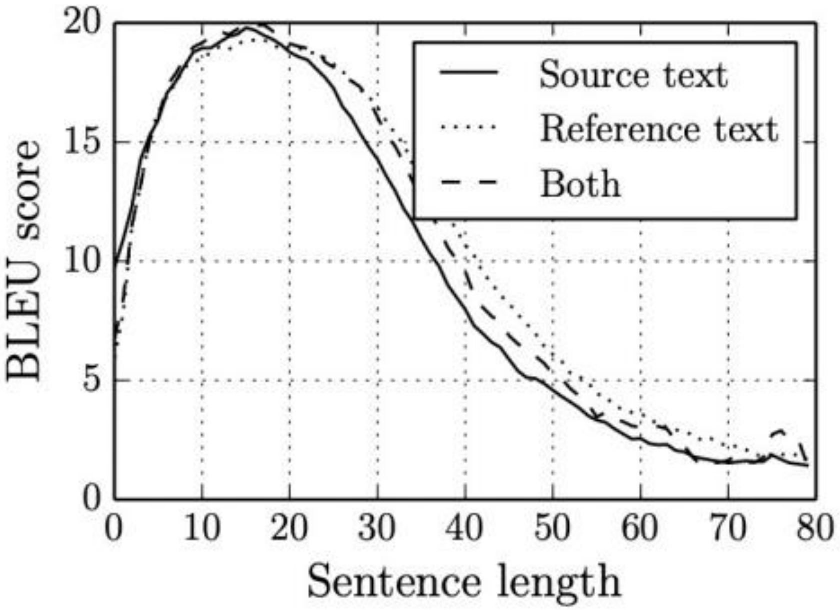


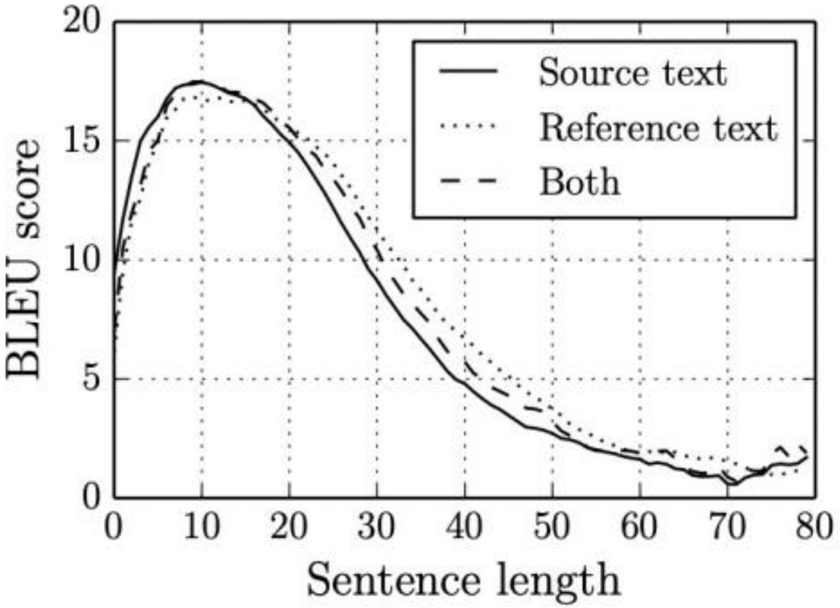


What performance trend is observed for inputs (source) and outputs (reference) as the number of words in each sentence grows?

Task: Machine Translation

(larger scores are better)

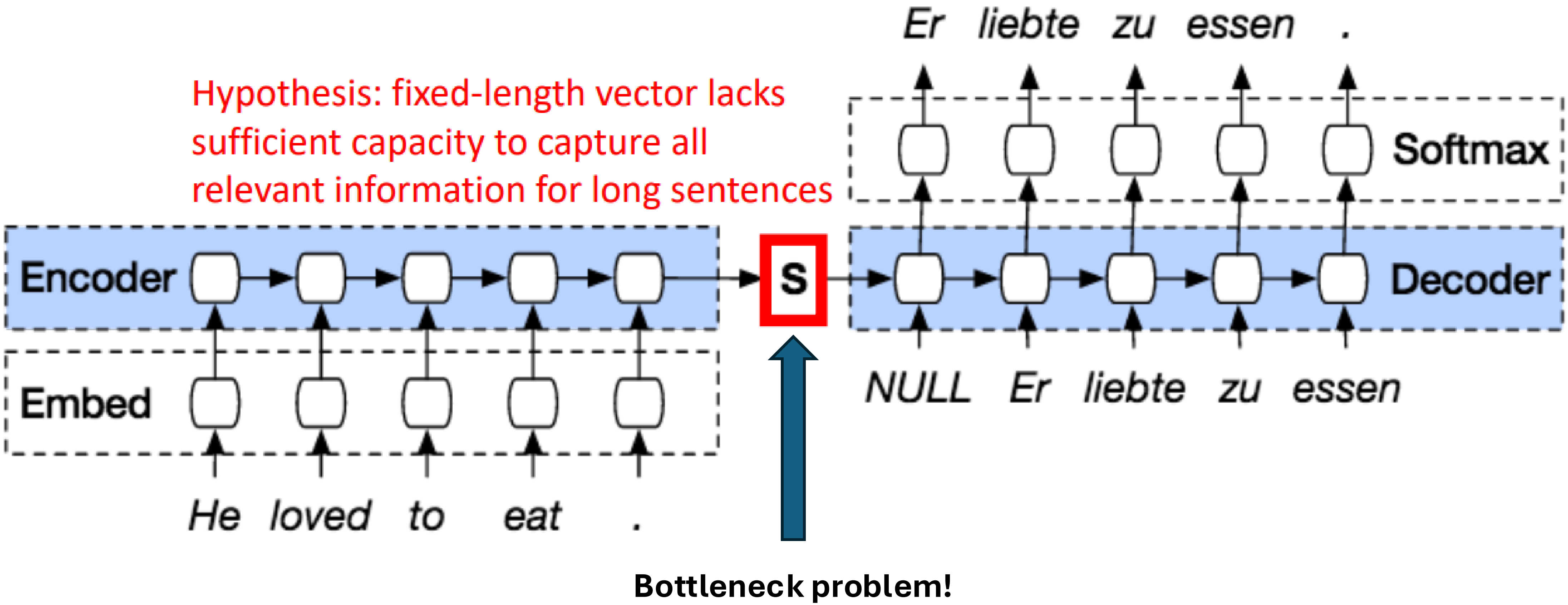




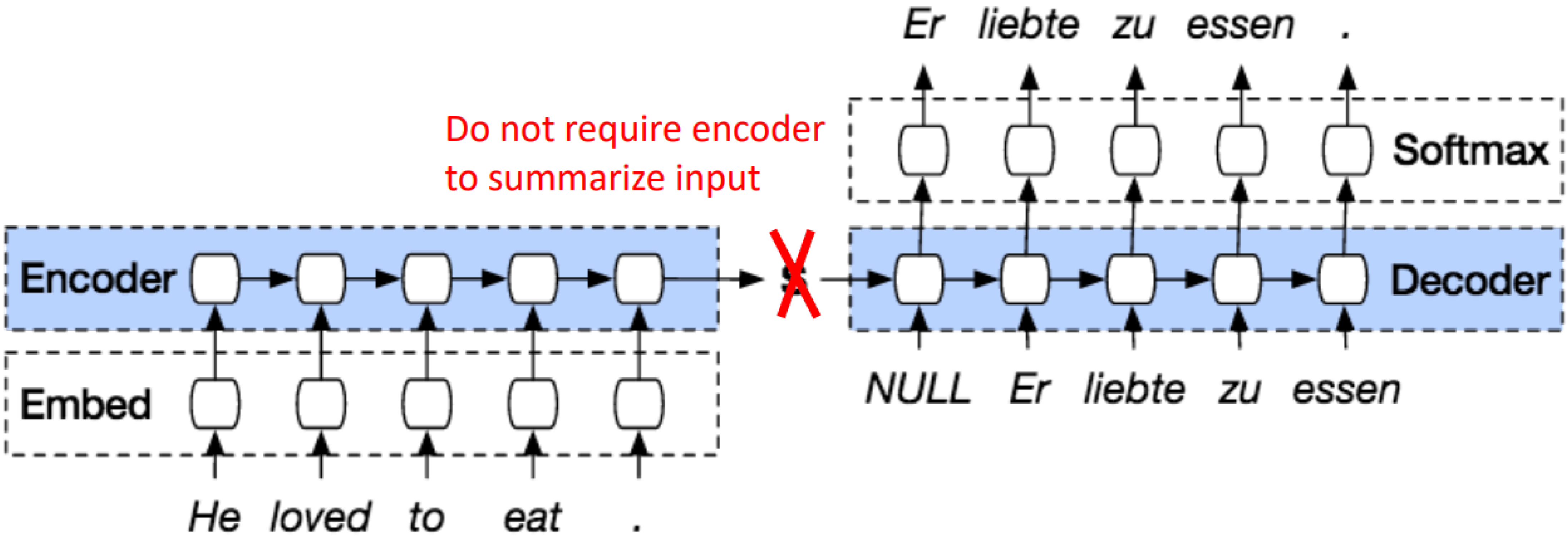
Performance drops for longer sentences!

Problem: Performance Drops as Sentence Length Grows

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

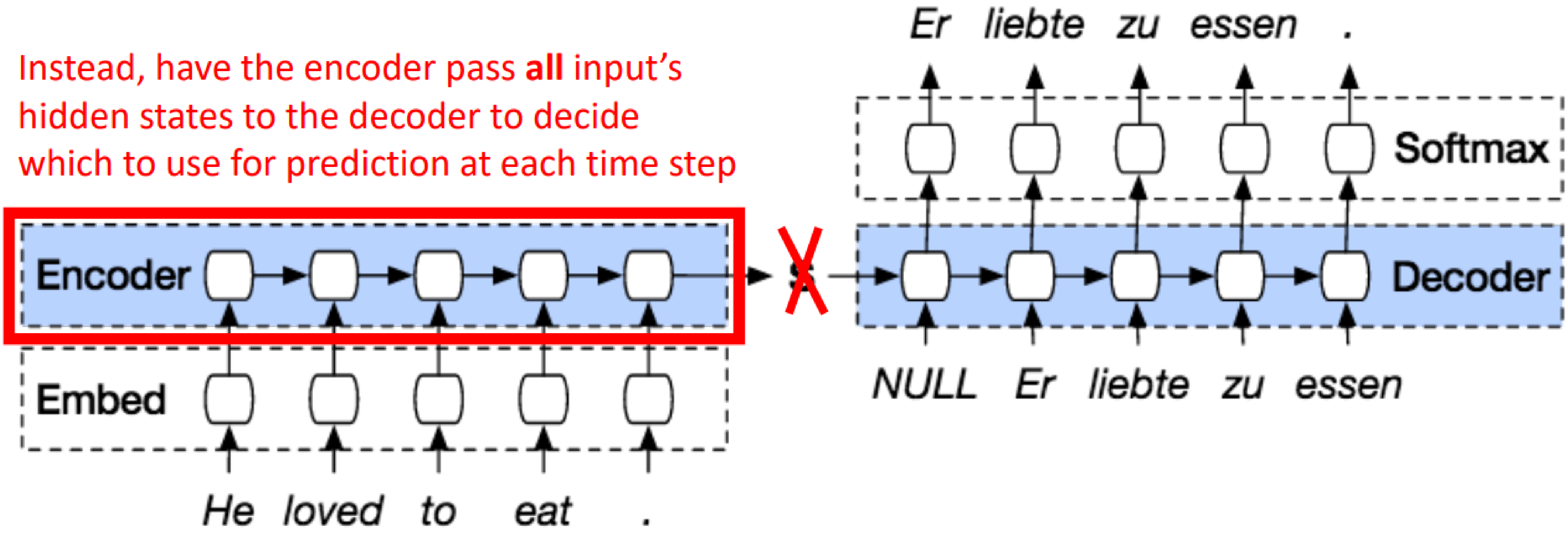


How to preserve Performance for Long Sequences?

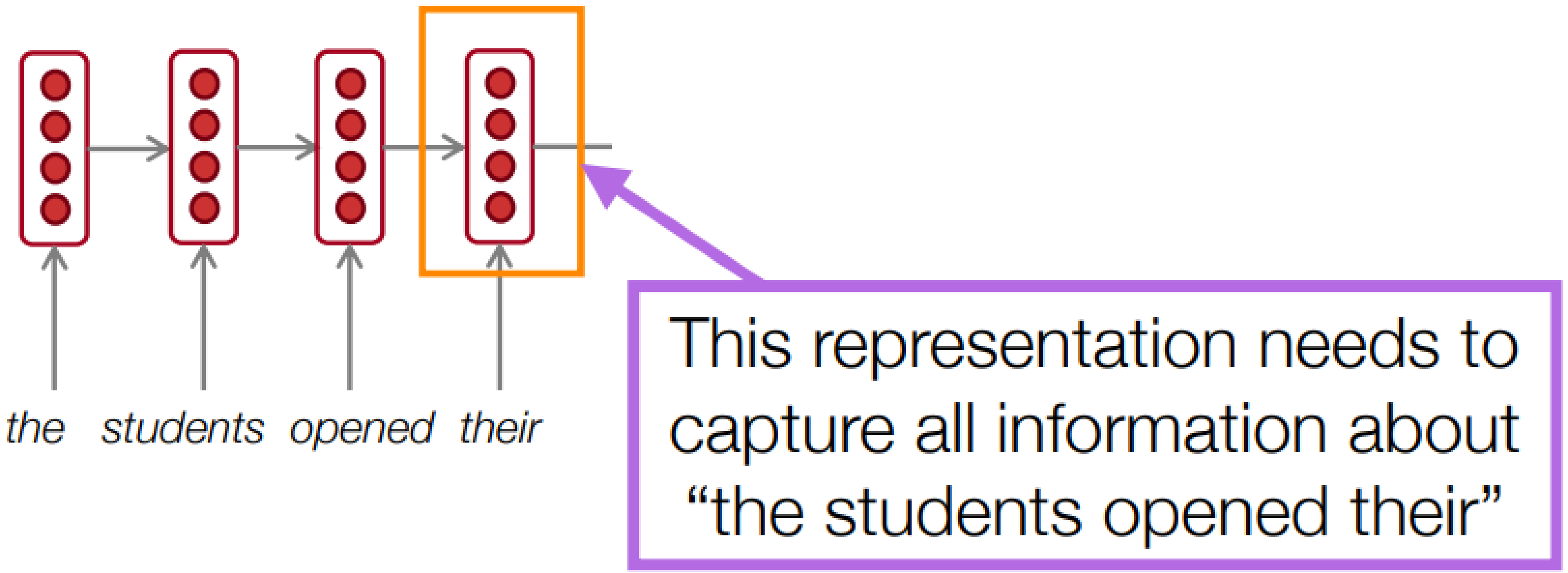


How to preserve Performance for Long Sequences?

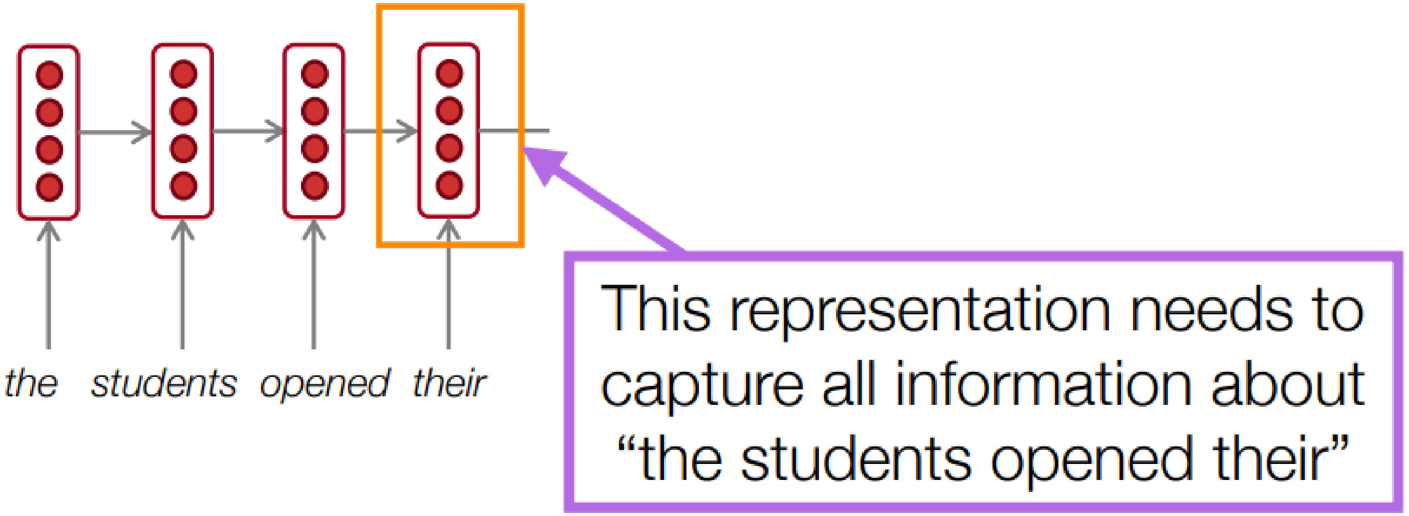
Instead, have the encoder pass **all** input's hidden states to the decoder to decide which to use for prediction at each time step



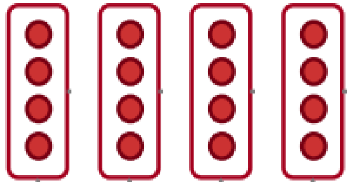
Idea: What If We Use Multiple Vectors?



Idea: What If We Use Multiple Vectors?



Instead of this, let's try:

the students opened their =  (all 4 hidden states!)

The Solution: Attention

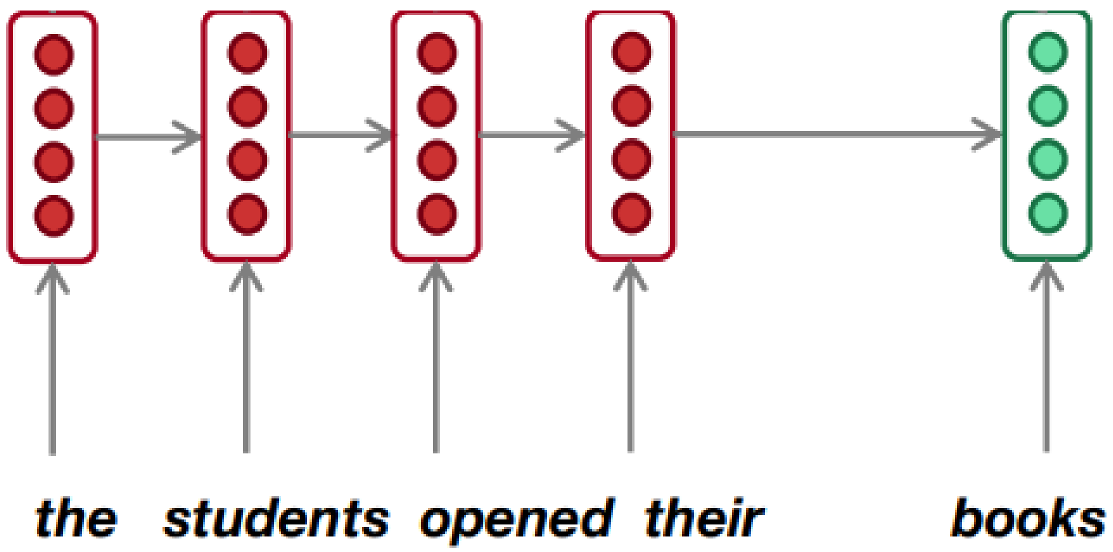
- **Attention mechanisms** (Bahdanau et al., 2015) allow language models to focus on a particular part of the observed context at each time step.
  - Originally developed for machine translation, and intuitively similar to word alignments between different languages

How does attention work?

- In general, we have a single **query vector** and multiple **key vectors**. We want to **score each query-key pair**.

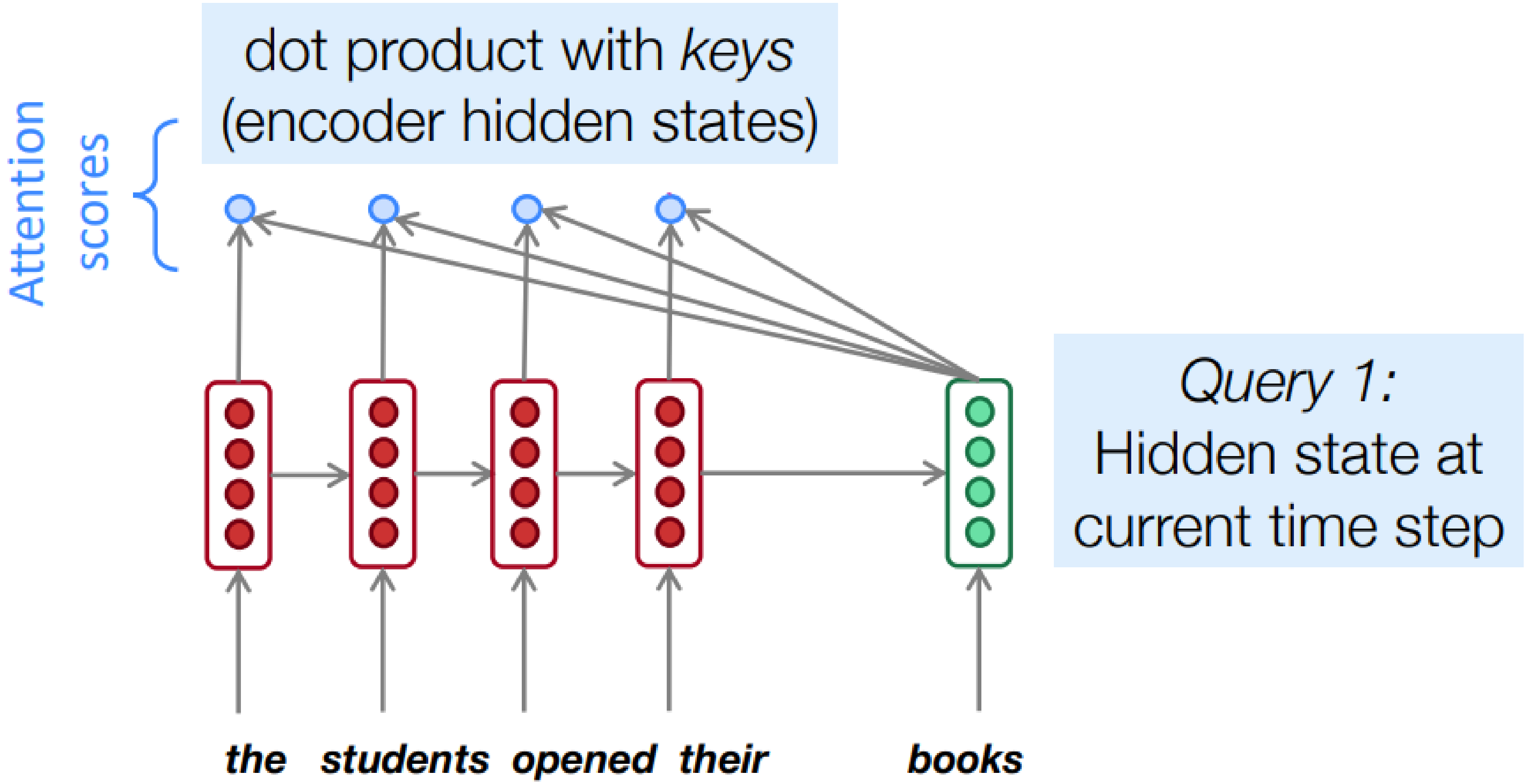
- In a neural language model, what are the queries and keys?

Attention Mechanisms in Neural Language Models

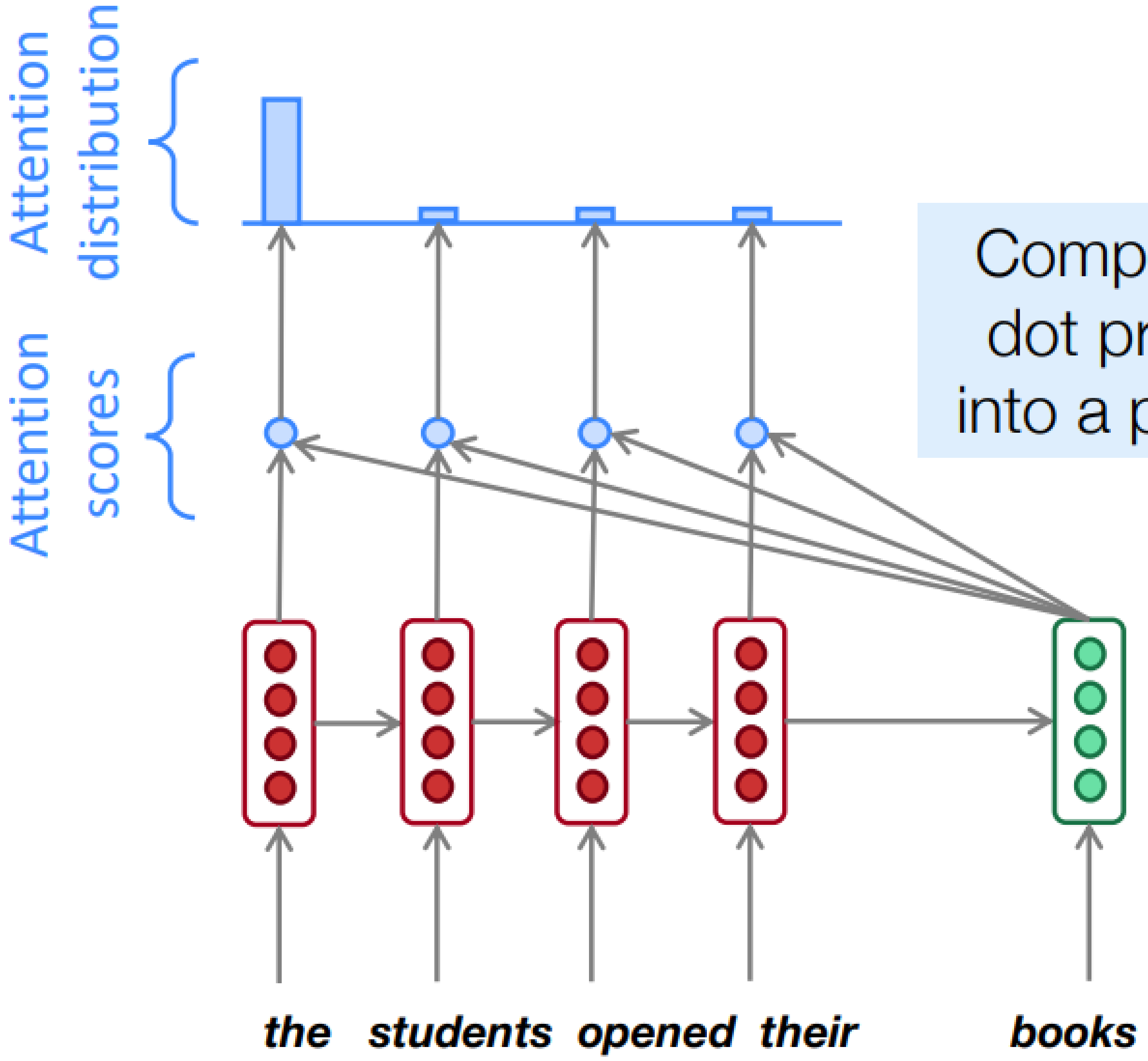


Query 1:
Hidden state at current time step

Attention Mechanisms in Neural Language Models



Attention Mechanisms in Neural Language Models



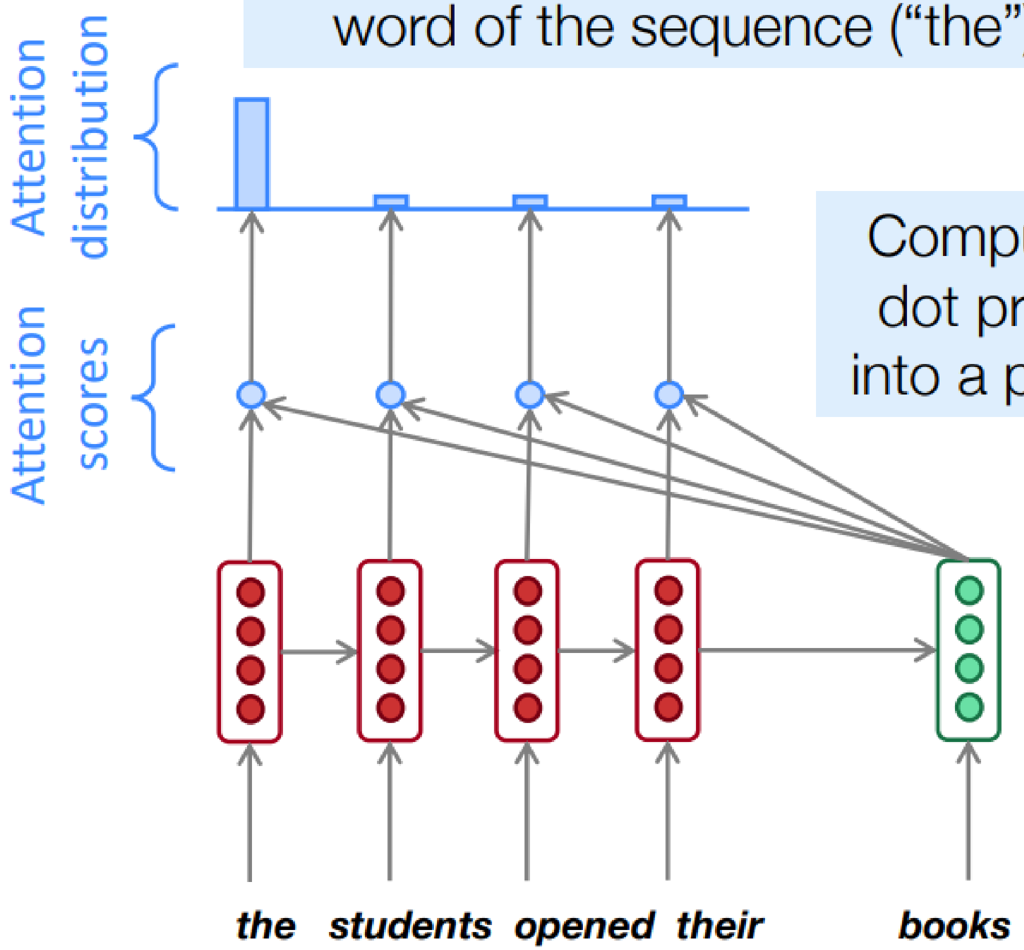
Compute softmax over the dot products to turn them into a probability distribution

Attention Mechanisms in Neural Language Models

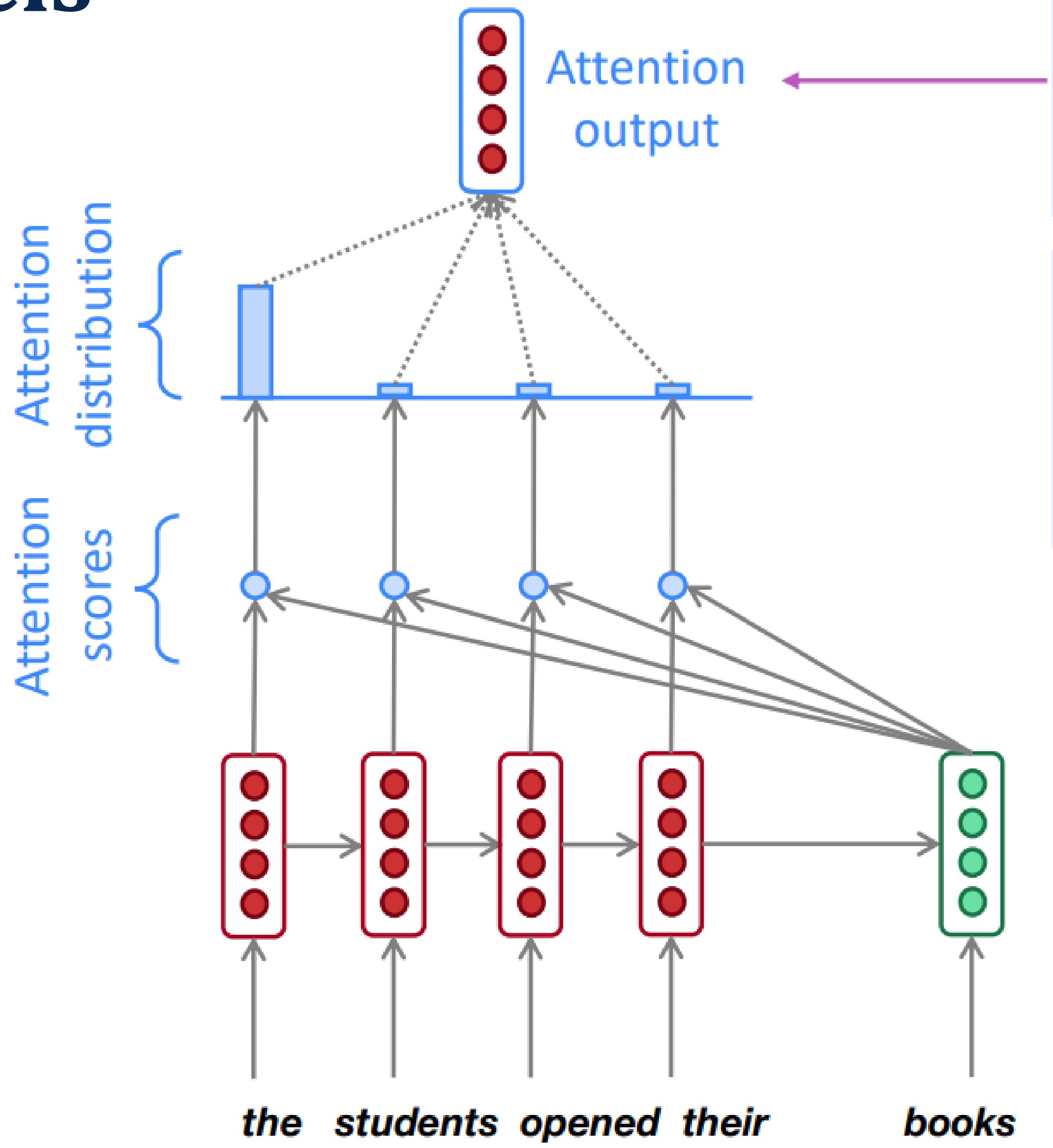
UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

At this time step, the attention distribution is focused on the first word of the sequence (“the”)

Compute softmax over the dot products to turn them into a probability distribution



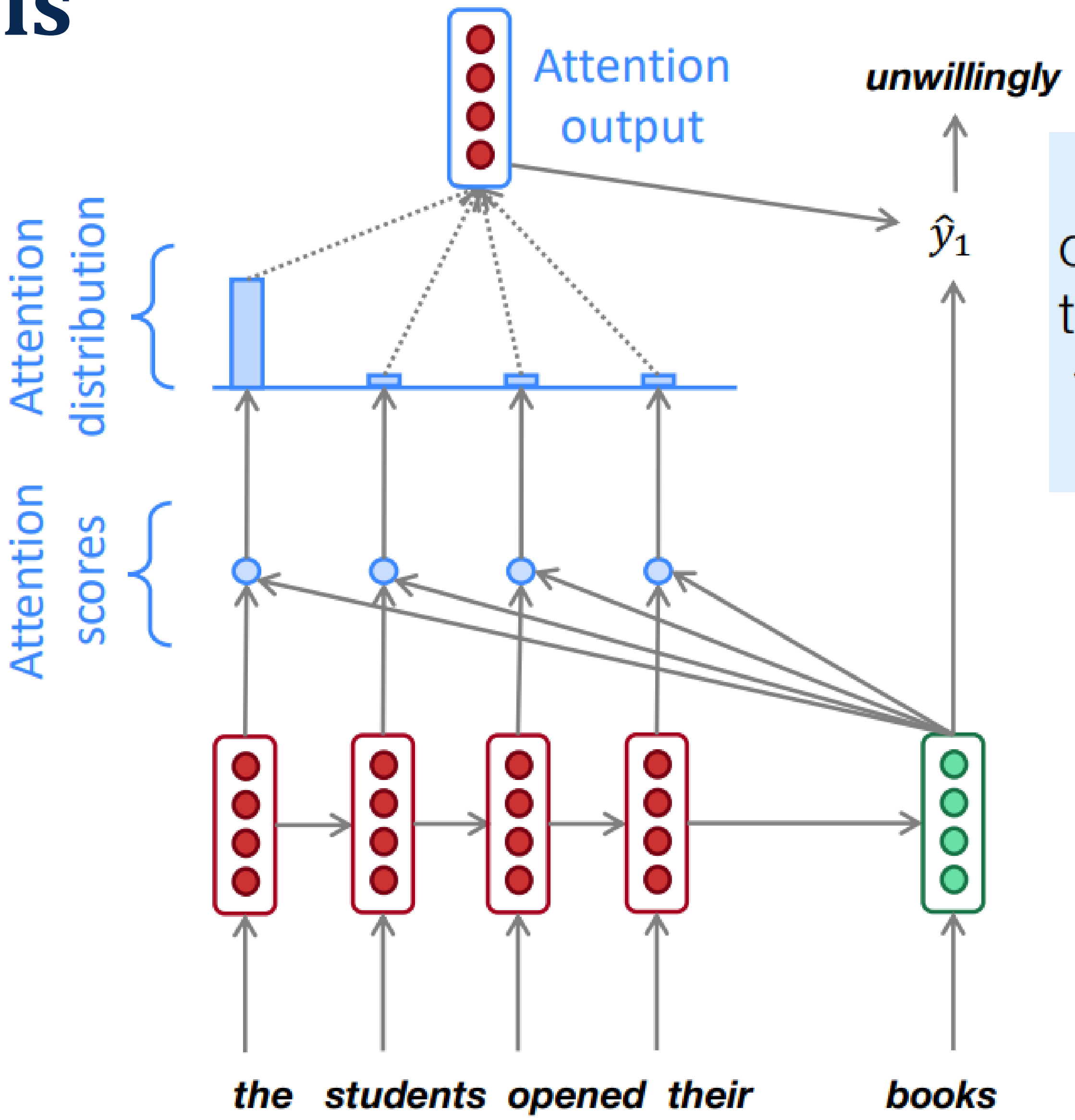
Attention Mechanisms in Neural Language Models



We use the attention distribution to compute a weighted average of the hidden states.

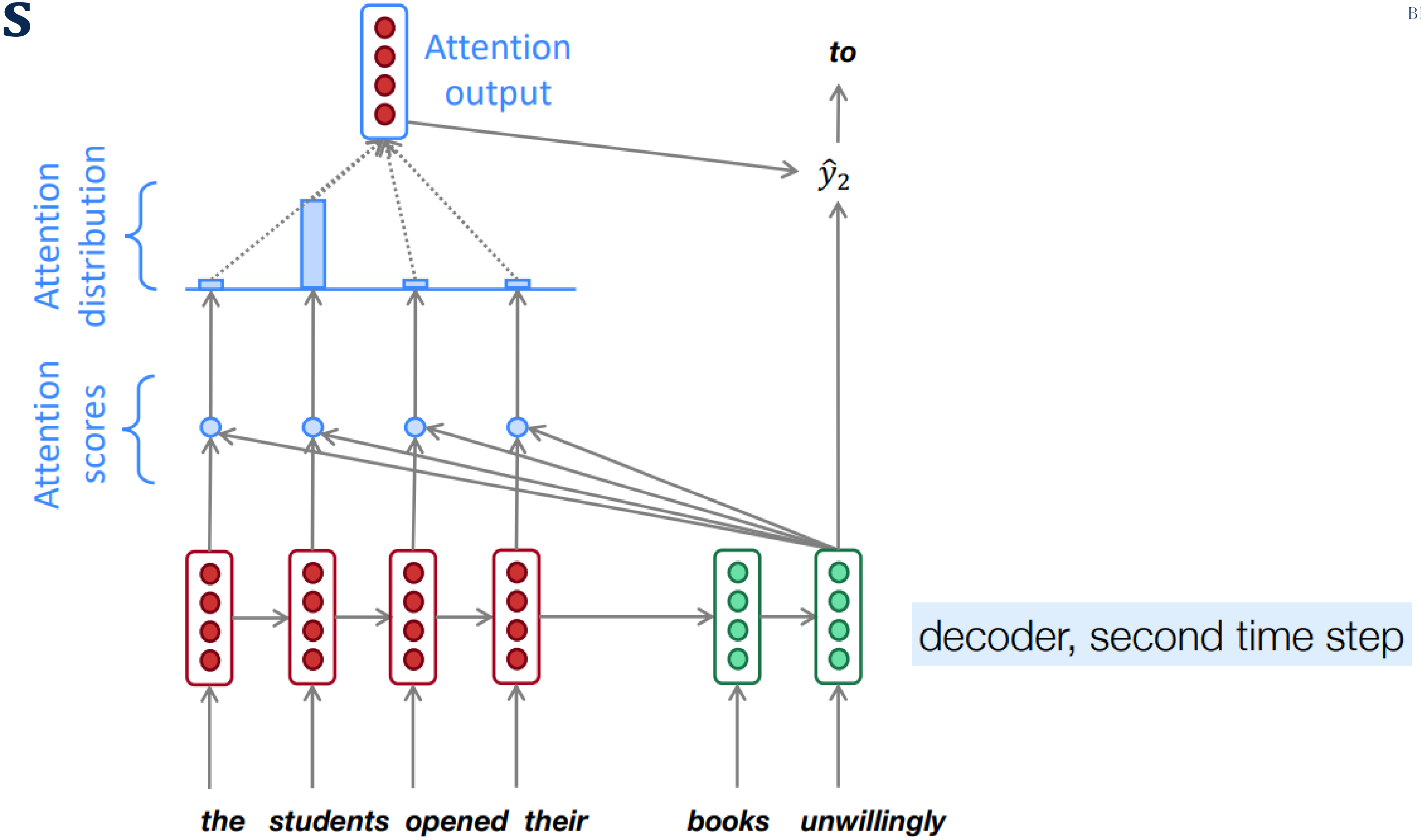
Intuitively, the resulting attention output contains information from hidden states that received high attention scores

Attention Mechanisms in Neural Language Models



Concatenate (or otherwise compose) the attention output with the current hidden state, then pass through a softmax layer to predict the next word

Attention Mechanisms in Neural Language Models



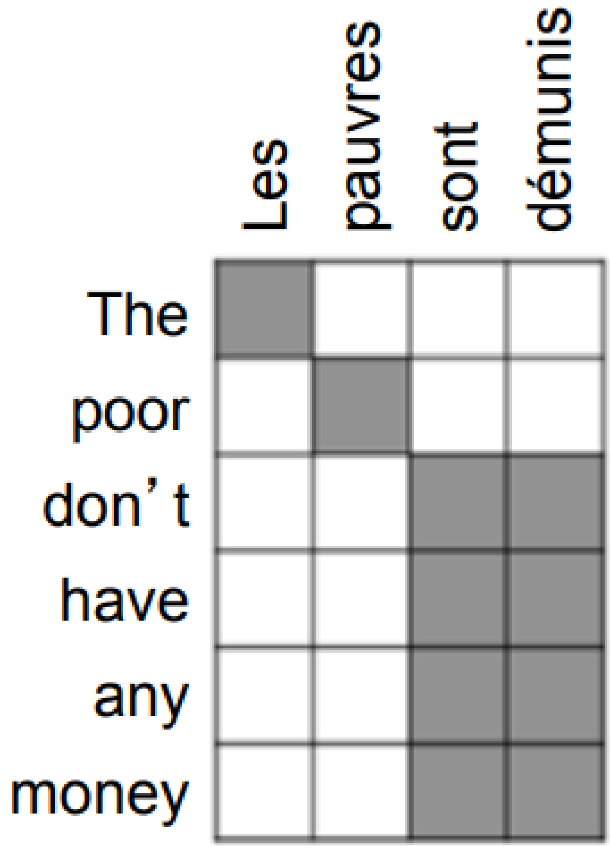
Attention Mechanisms in Neural Language Models

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

- Attention **solves the bottleneck problem**
  - Attention allows decoder to look directly at source; bypass bottleneck

- Attention **helps with vanishing gradient problem**
  - Provides shortcut to faraway states (???)

- Attention provides some **interpretability**
  - By inspecting attention distribution, we can see what the decoder was focusing on
  - We get an alignment for free!
  - This is cool because we never explicitly trained an alignment system
  - The network just learned alignment by itself



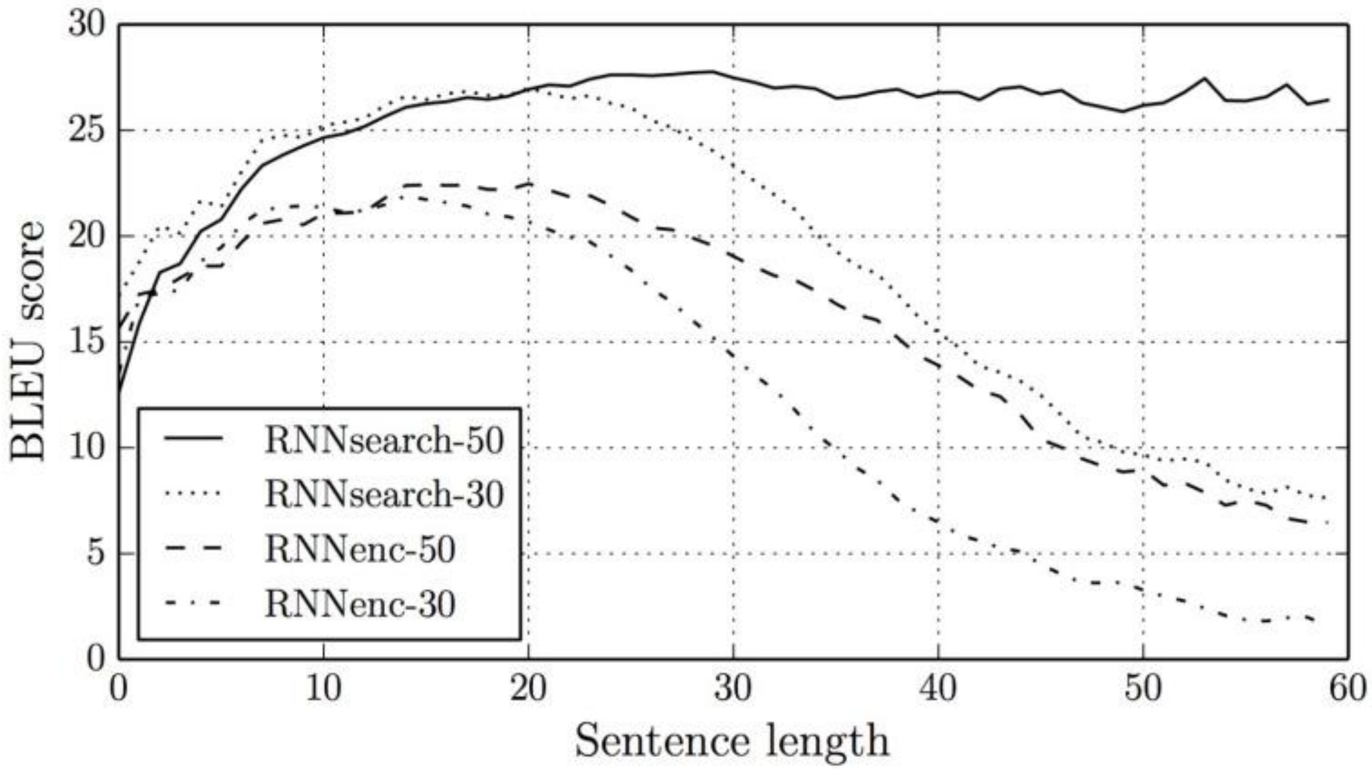
Many Variants of Attention

- Original formulation: $a(\mathbf{q}, \mathbf{k}) = w_2^T \tanh(W_1[\mathbf{q}; \mathbf{k}])$
- Bilinear product: $a(\mathbf{q}, \mathbf{k}) = \mathbf{q}^T W \mathbf{k}$ Luong et al., 2015
- Dot product: $a(\mathbf{q}, \mathbf{k}) = \mathbf{q}^T \mathbf{k}$ Luong et al., 2015
- Scaled dot product: $a(\mathbf{q}, \mathbf{k}) = \frac{\mathbf{q}^T \mathbf{k}}{\sqrt{|\mathbf{k}|}}$ Vaswani et al., 2017

Analysis of Attention Models

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

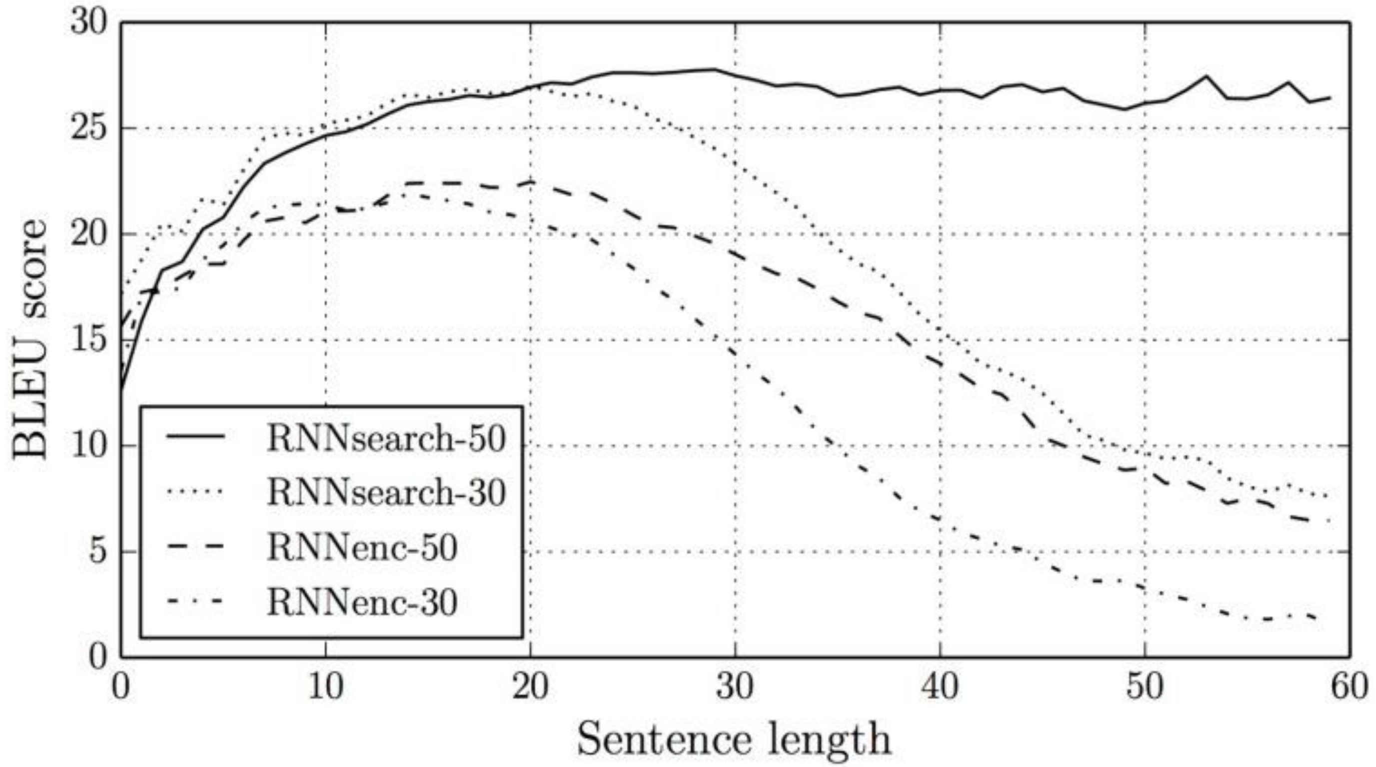
(larger scores are better)



What performance trend is observed as the number of words in the input sentence grows?

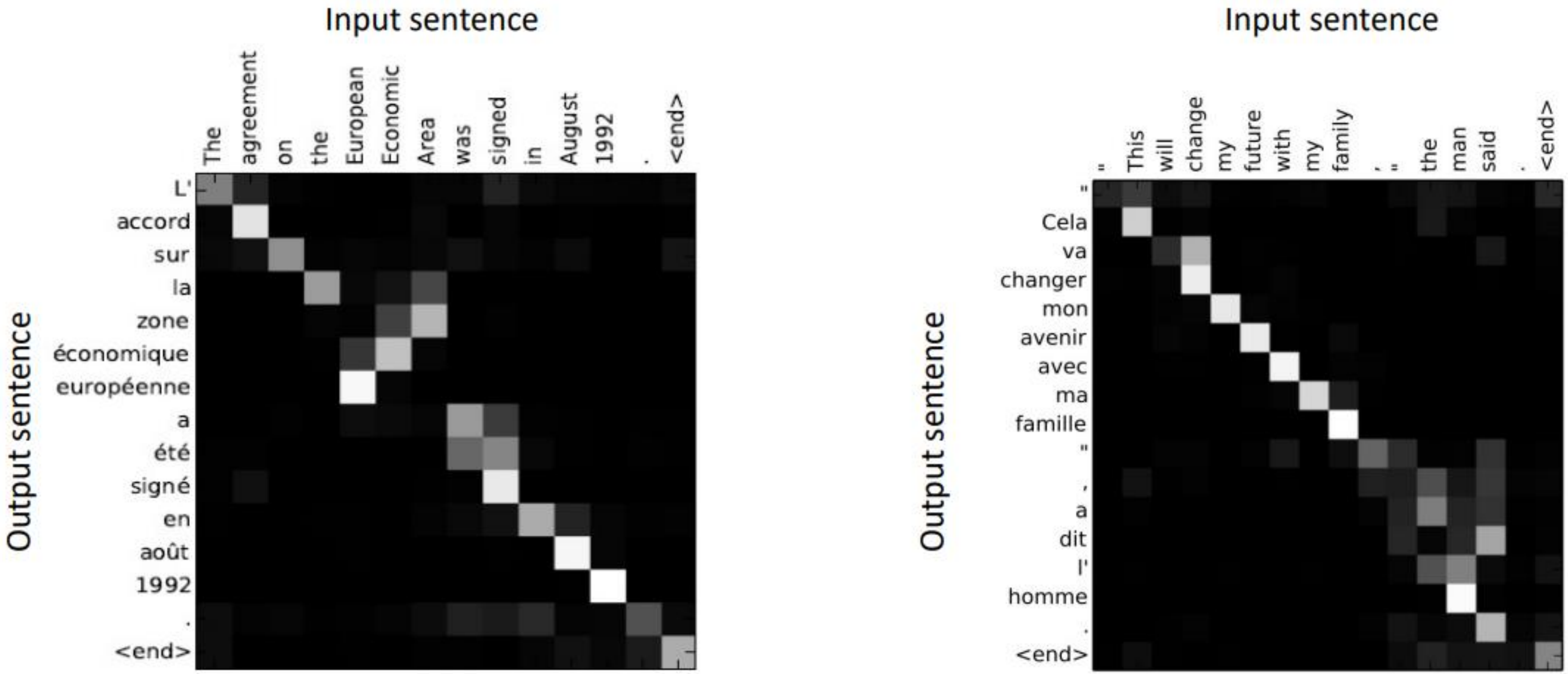
Analysis of Attention Models

(larger scores are better)



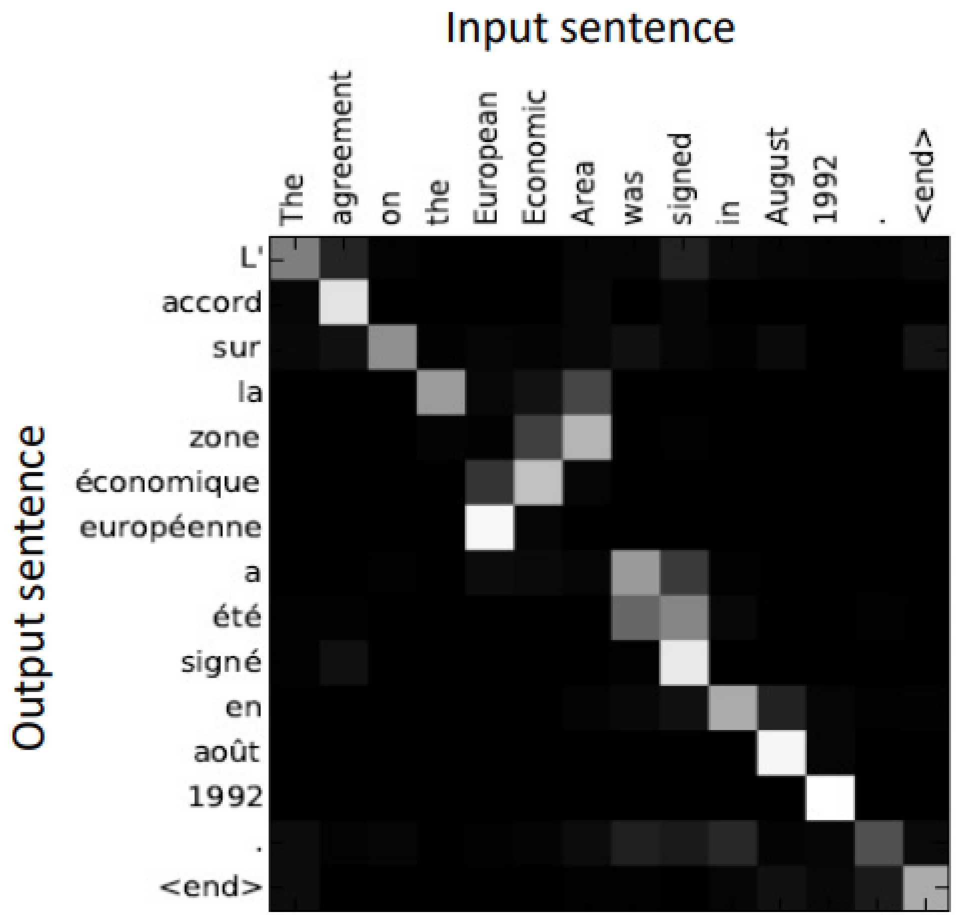
Performance no longer drops for longer sentences!

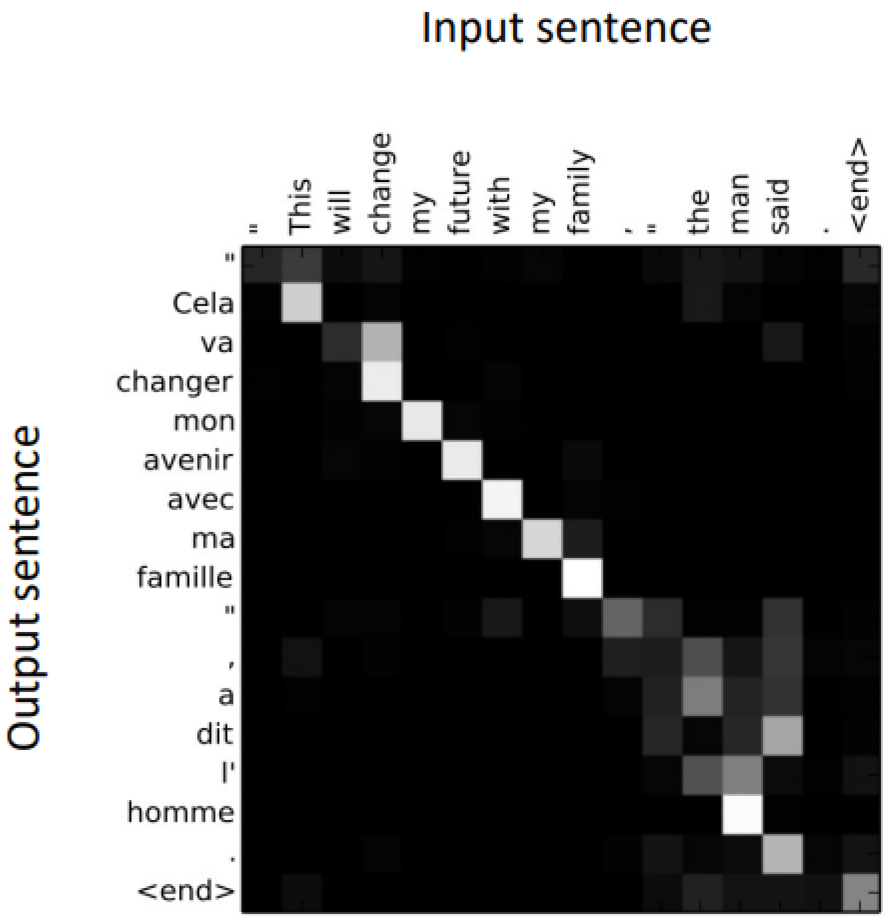
Visualizing Attention



Values are 0 to 1, with whiter pixels indicating larger attention weights

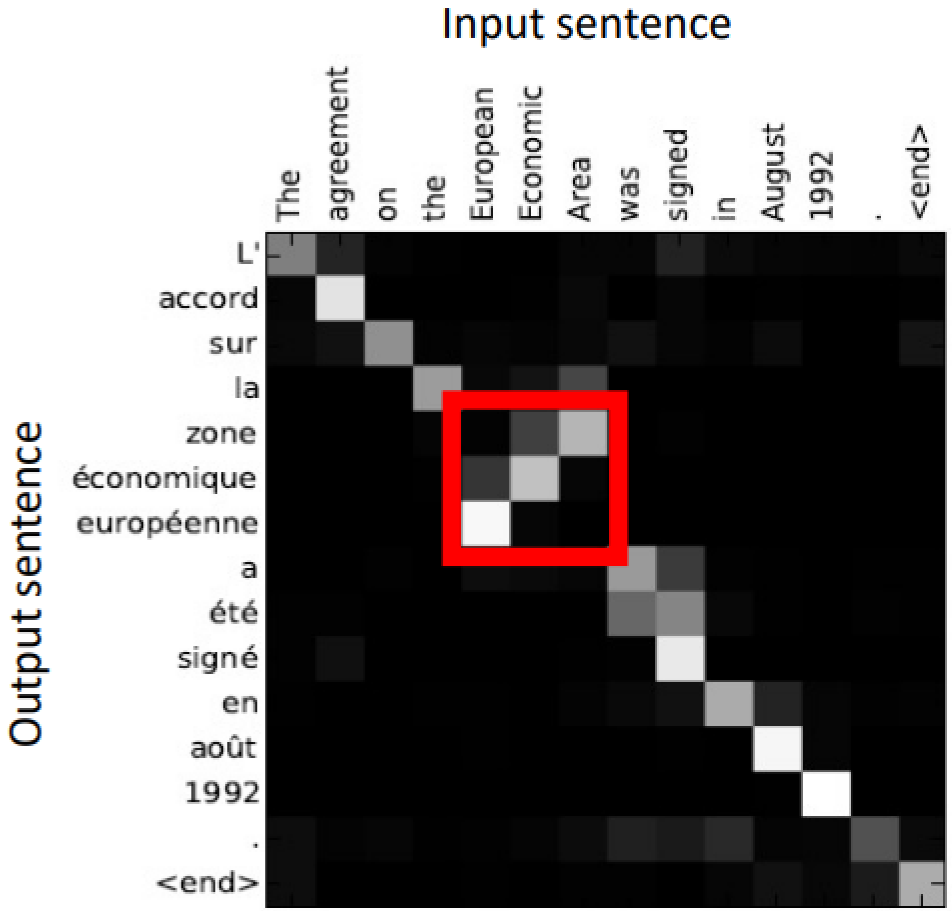
Visualizing Attention

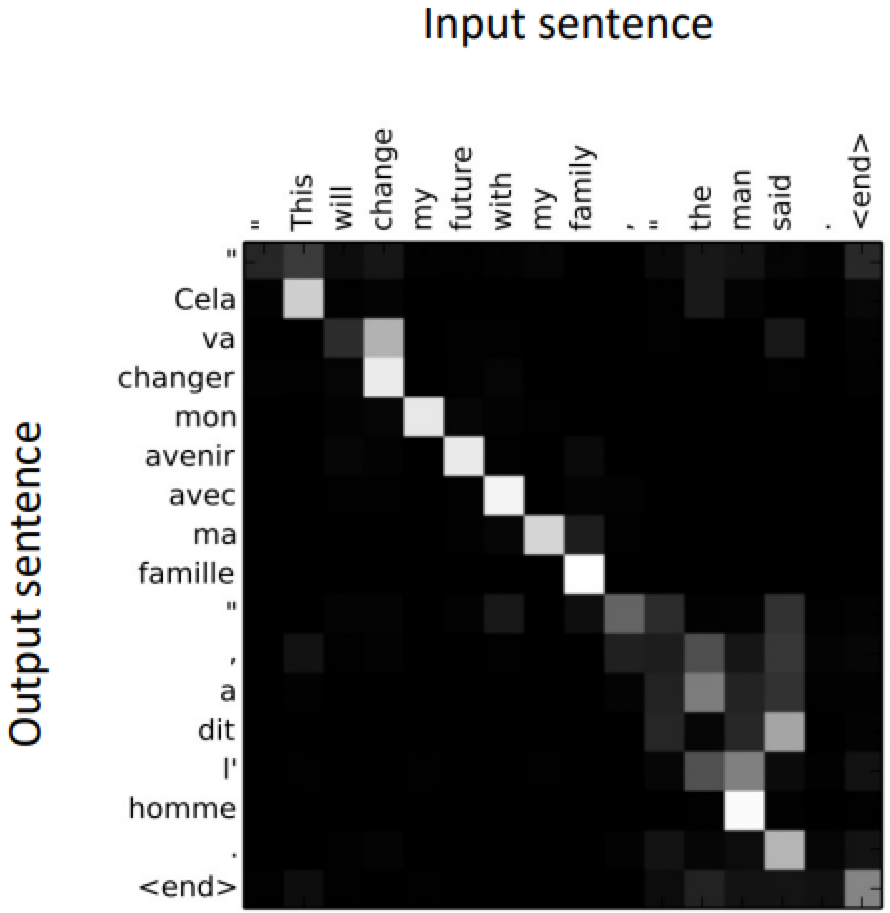




What insights can we glean from these examples?

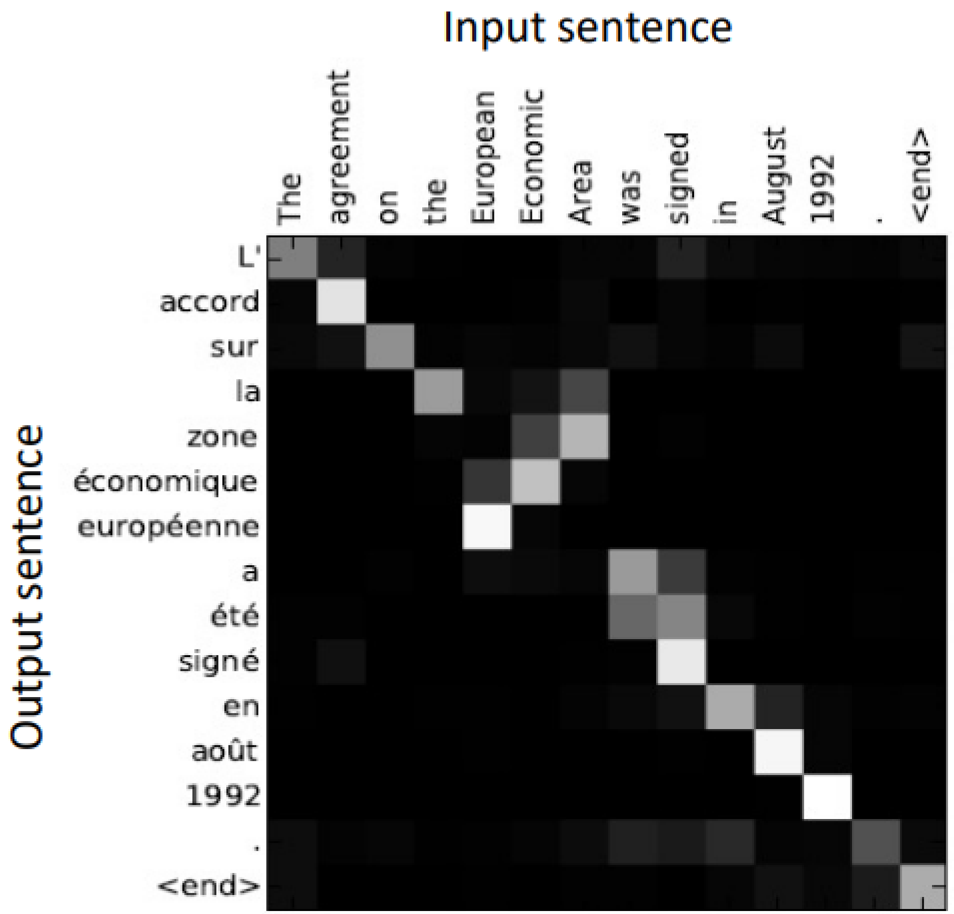
Visualizing Attention

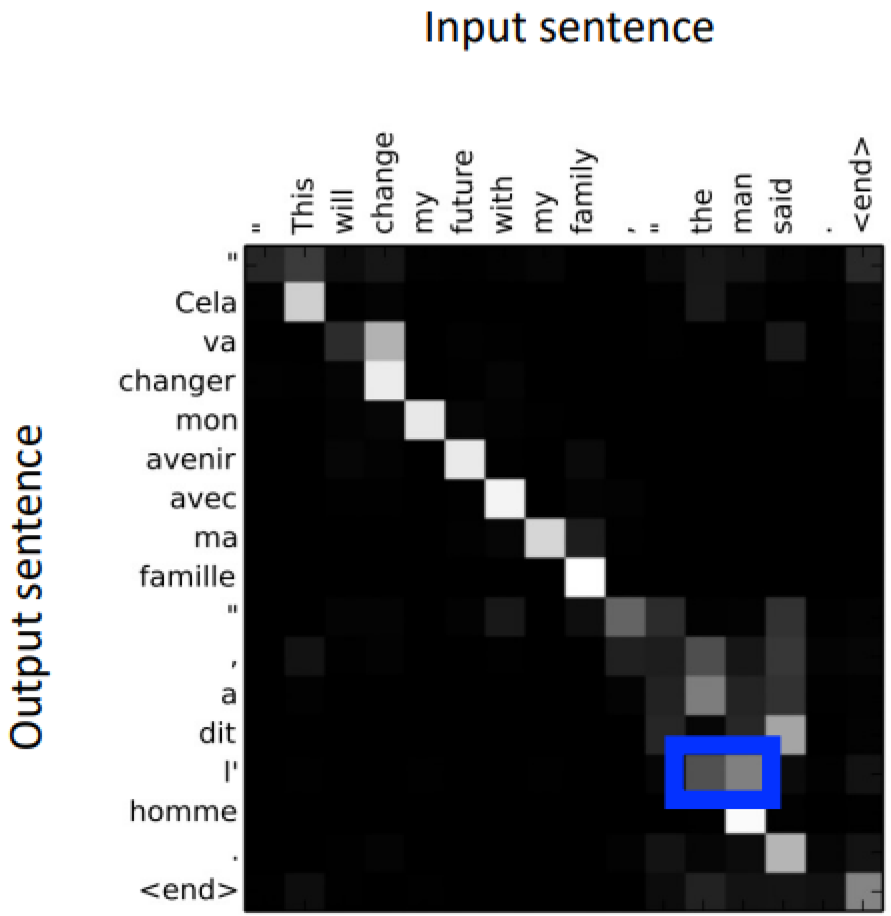




While a linear alignment between input and output sentences is common, there are exceptions (e.g., order of adjectives and nouns can differ)

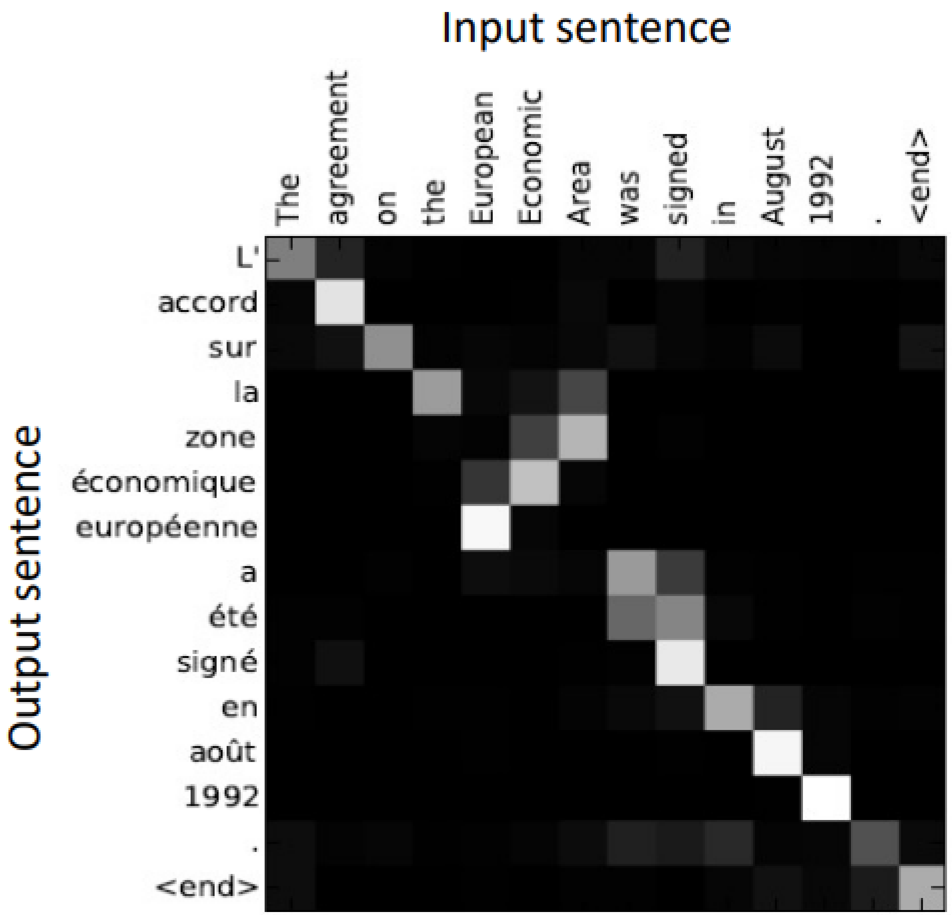
Visualizing Attention

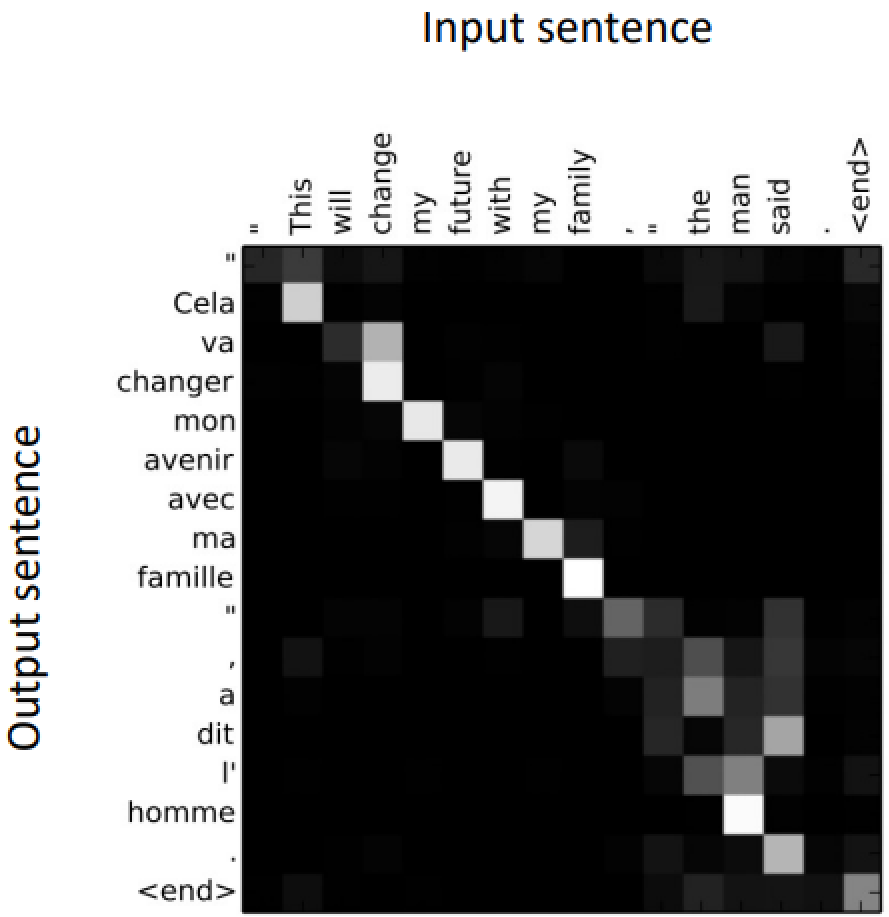




Output words are often informed by more than one input word;
e.g., “man” indicates translation of “the” to l’ instead of le, la, or les

Visualizing Attention





It naturally handles different input and output lengths
(e.g., 1 extra output word for both examples)

Exercises

- Let's implement a attention mechanisms with PyTorch.

- Follow the instructions on the notebook: "seq2seq_translation_exercises.ipynb".